Deploying Dynamic Analyses and Preventing Compiler Backdoors with Multi-Version Execution

Luís Pina[Imperial College London]
l.pina@imperial.ac.uk

Joint work with Cristian Cadar[Imperial College London], Anastasios Andronidis[Imperial College London], and John Regehr[U]

[Imperial College London] Imperial College London, UK
[U] University of Utah, USA

Runtime Verification beyond Monitoring (ArVi)
ICT COST Action IC1402

Barcelona, March 10th, 2016

Deploying Dynamic Analysis?

Why?

```
./server
```

Deploying Dynamic Analysis?

Why?

```
./server
```

```
Segmentation fault
```

Deploying Dynamic Analysis?

Why?

```
./server
```

Deploying Dynamic Analysis?

Why?

```
valgrind --tool=memcheck server
```

```
Invalid read of size 32
   by 0x40B07FF4: memcpy
                      (mc_replace_strmem.c:635)
   by 0x40AC751B: dtls1_process_heartbeat(SSL *s)
                      (ssl/d1_both.c:1497)
 Address 0xBFFFF0E0 is not stack'd, malloc'd or free'd
```

Deploying Dynamic Analysis?

Why?

```
valgrind --tool=memcheck server
```

```
Invalid read of size 32
   by 0x40B07FF4: memcpy
                      (mc_replace_strmem.c:635)
   by 0x40AC751B: dtls1_process_heartbeat(SSL *s)
                      (ssl/d1_both.c:1497)
 Address 0xBFFFF0E0 is not stack'd, malloc'd or free'd
```

7x–57x slowdown

Deploying Dynamic Analysis?

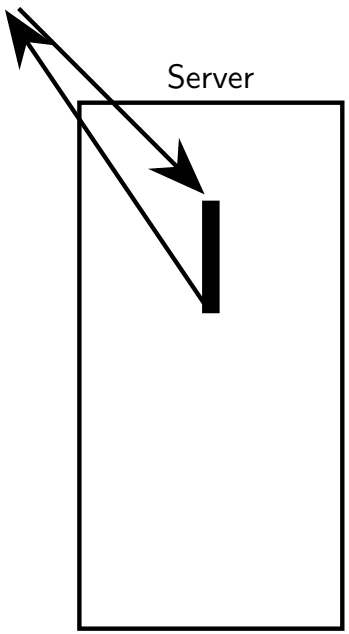
Why?

```
gcc -fsanitize=address server.c -o server
./server
```

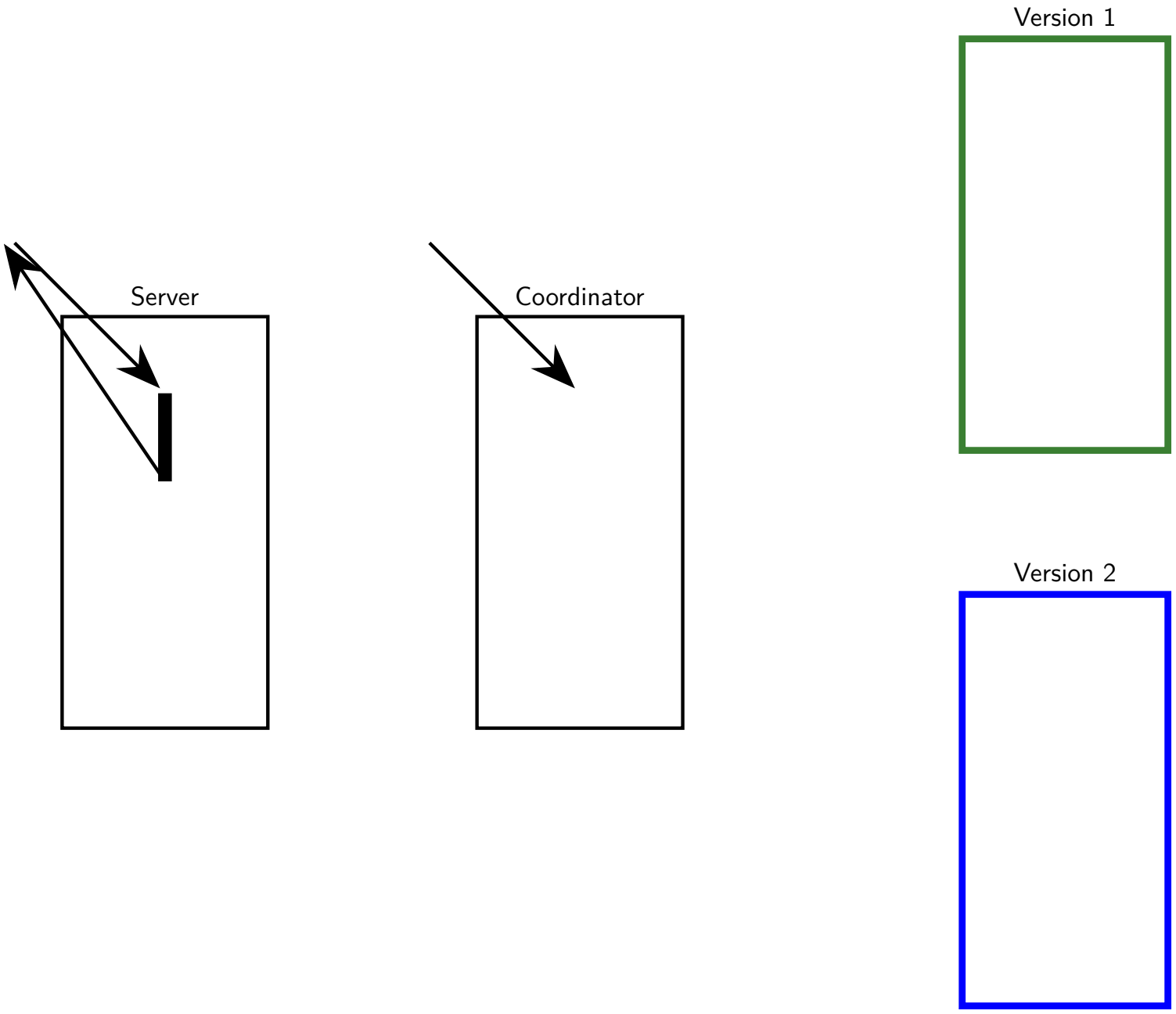
```
==2268==ERROR: AddressSanitizer: heap-buffer-overflow
       on address 0x629000013748 at pc 0x7f228f5f0cfa
READ of size 32768 at 0x629000013748 thread T0
   #0 0x43d075 in memcpy /usr/include/bits/string3.h:51
   #1 0x43d075 in tls1_process_heartbeat ssl/t1_lib.c:2586
   #2 0x50e498 in ssl3_read_bytes ssl/s3_pkt.c:1092
   #3 0x51895c in ssl3_get_message ssl/s3_both.c:457
...
==2268== ABORTING
```

1.10x–2.67x slowdown

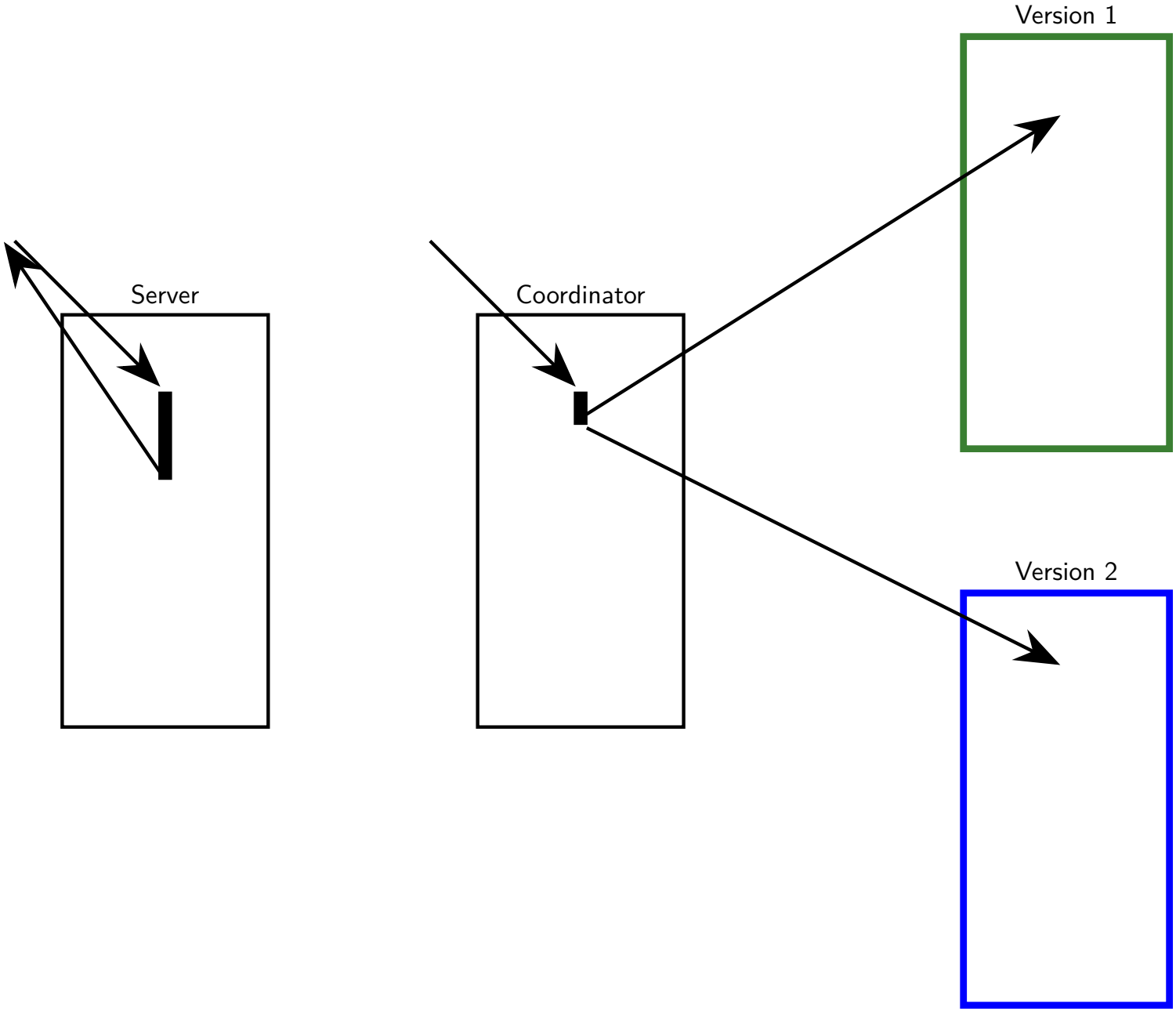
N-Version Execution



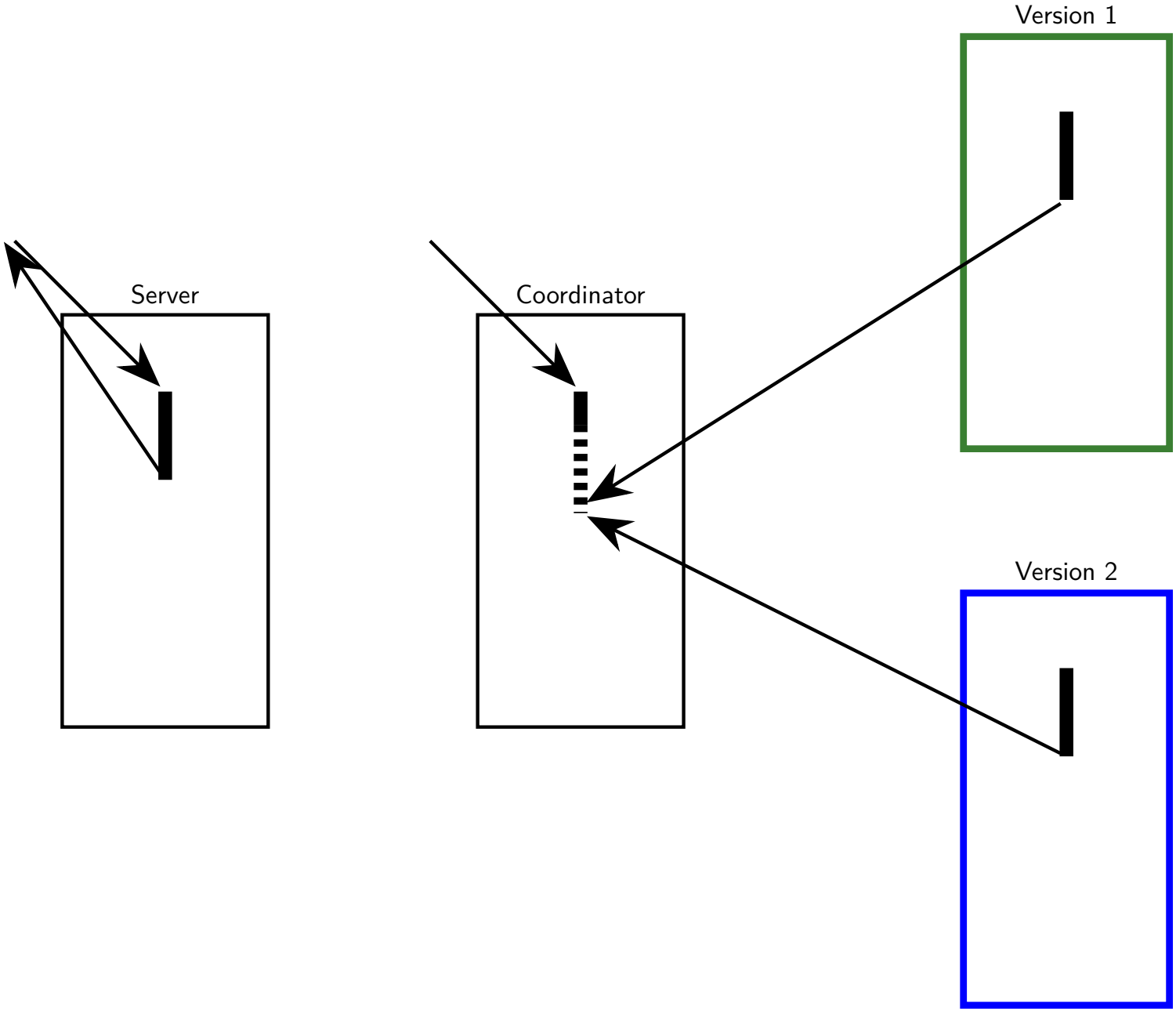
N-Version Execution



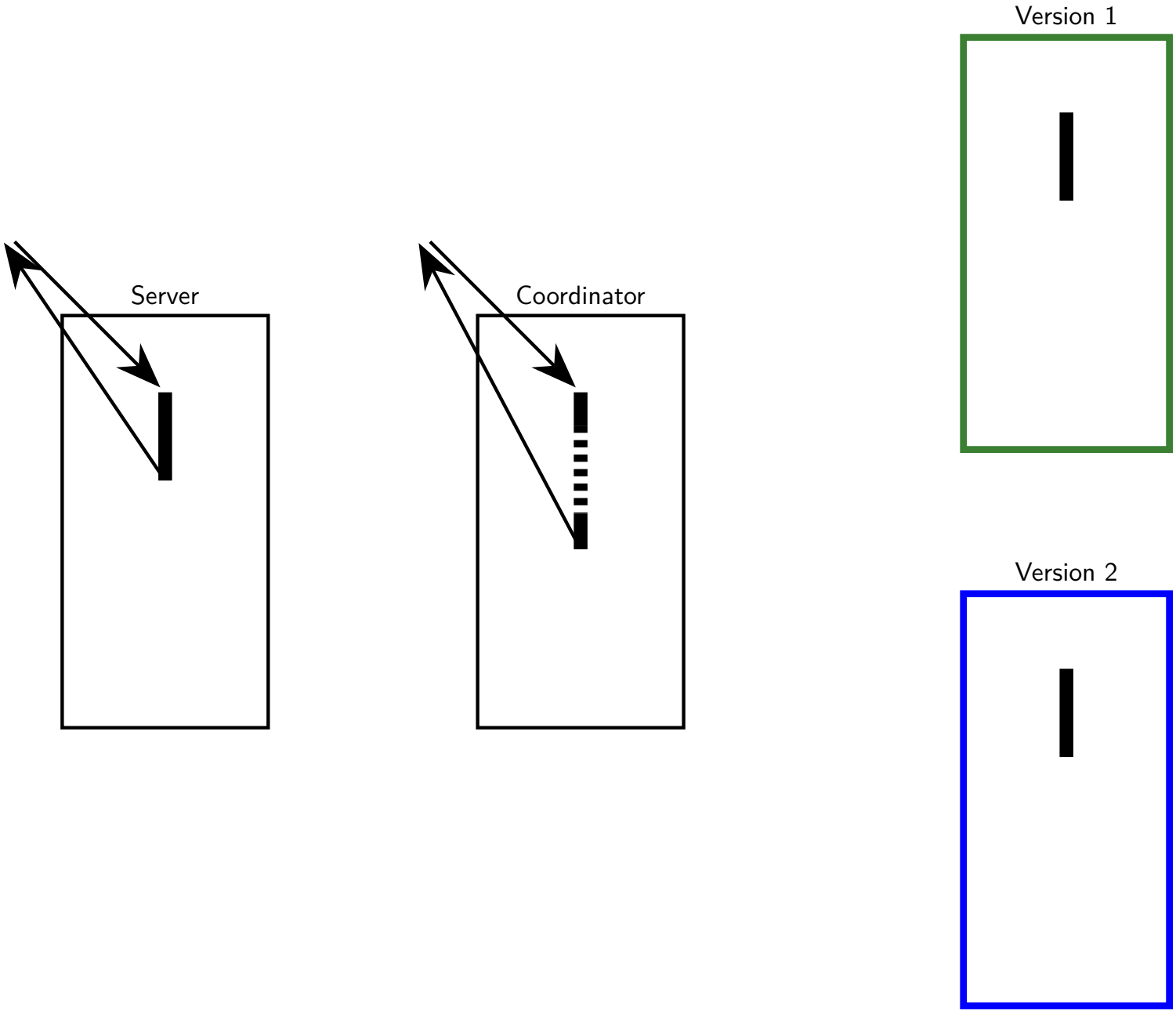
N-Version Execution



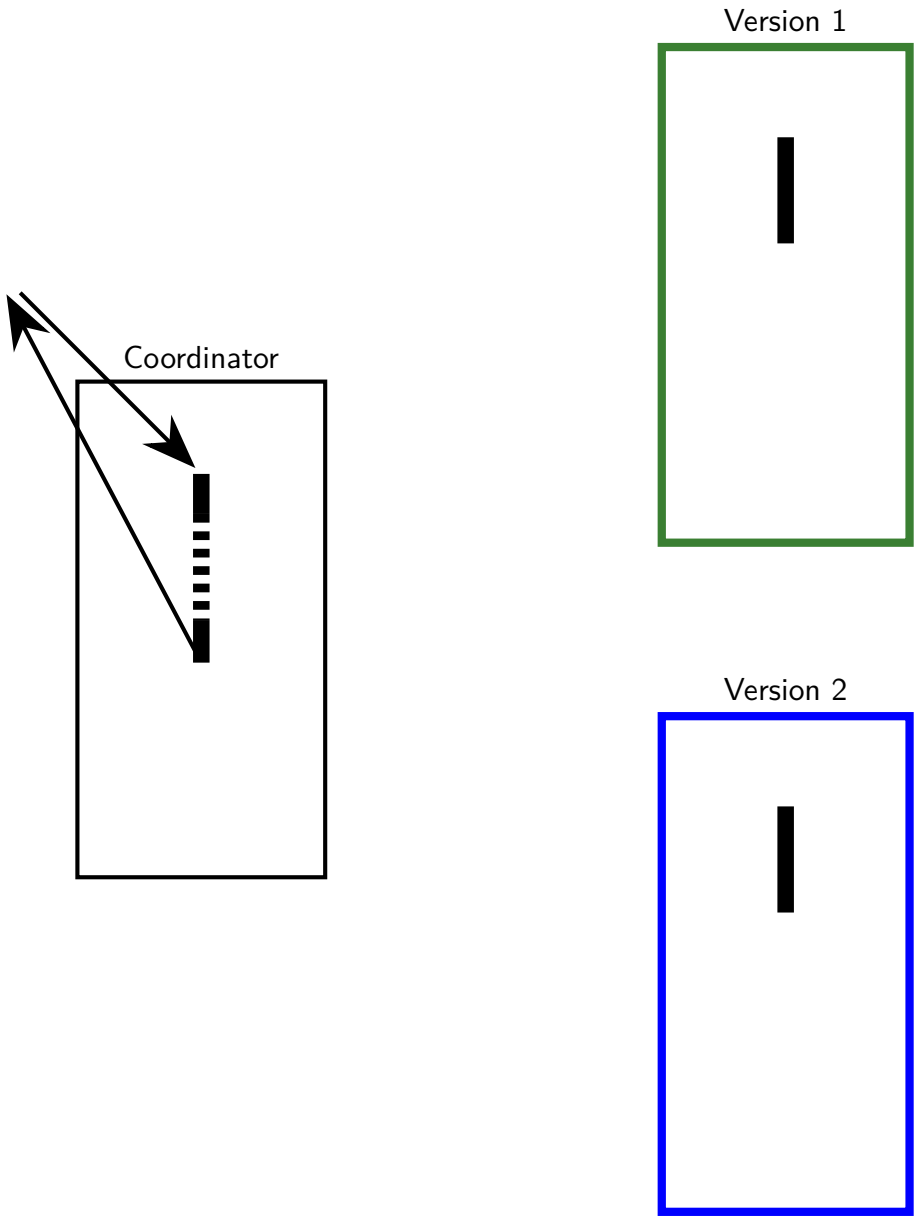
N-Version Execution



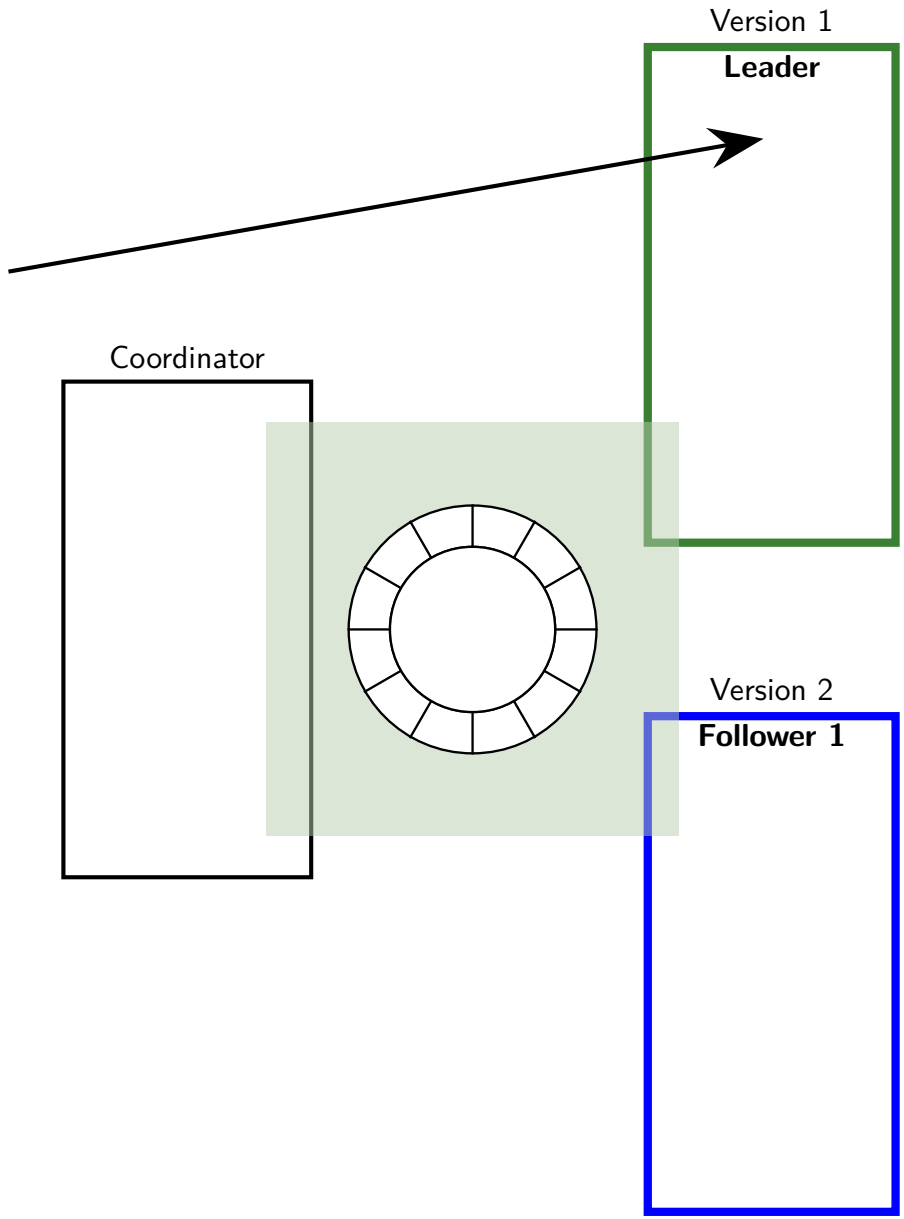
N-Version Execution



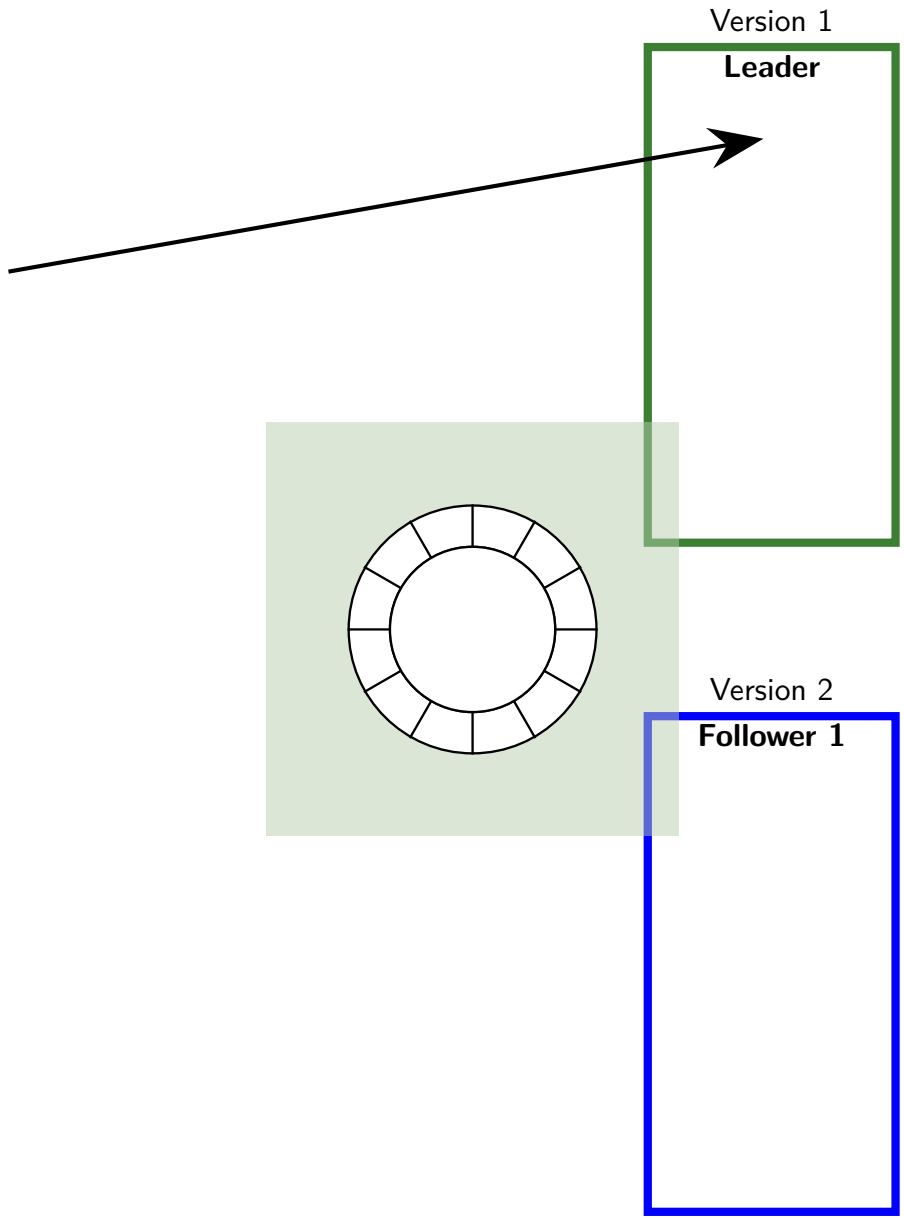
Varan



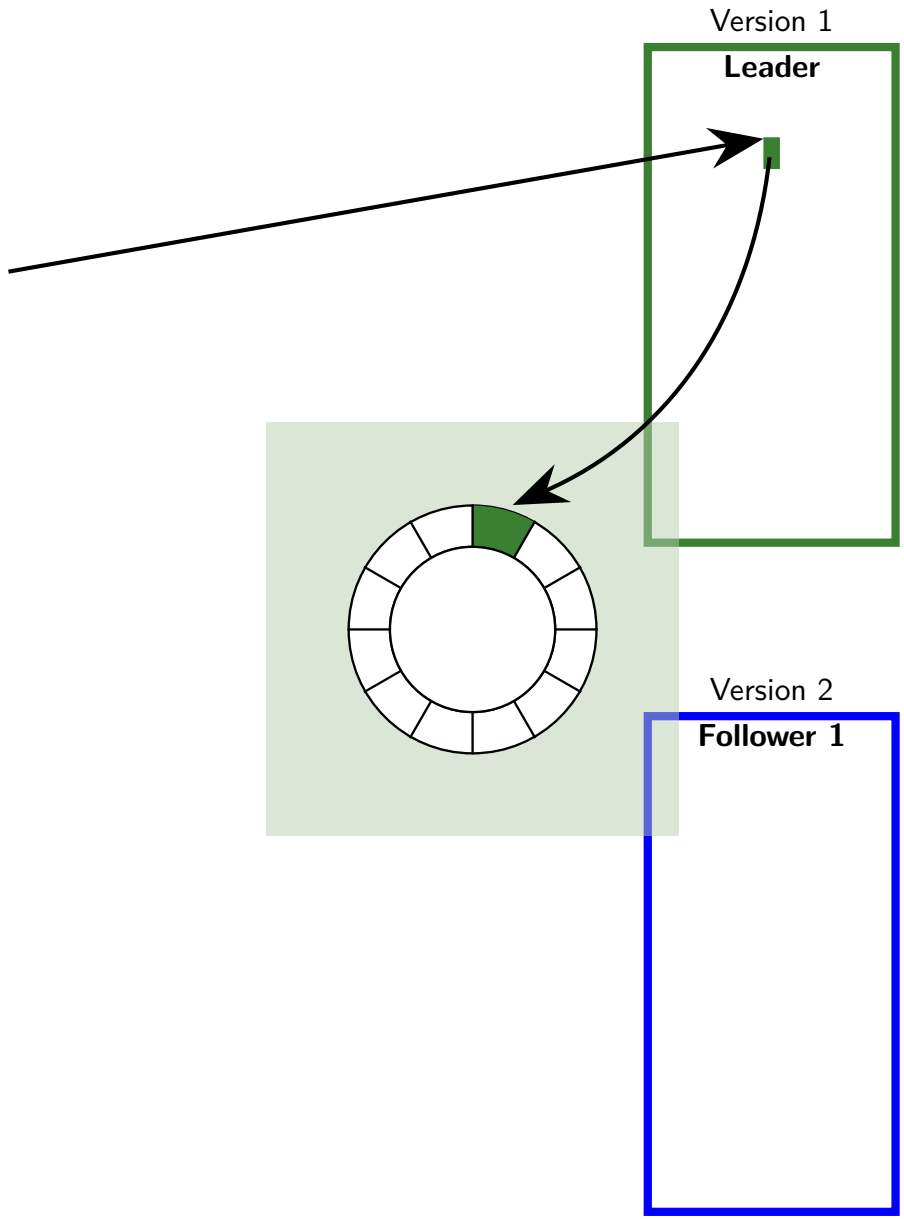
Varan



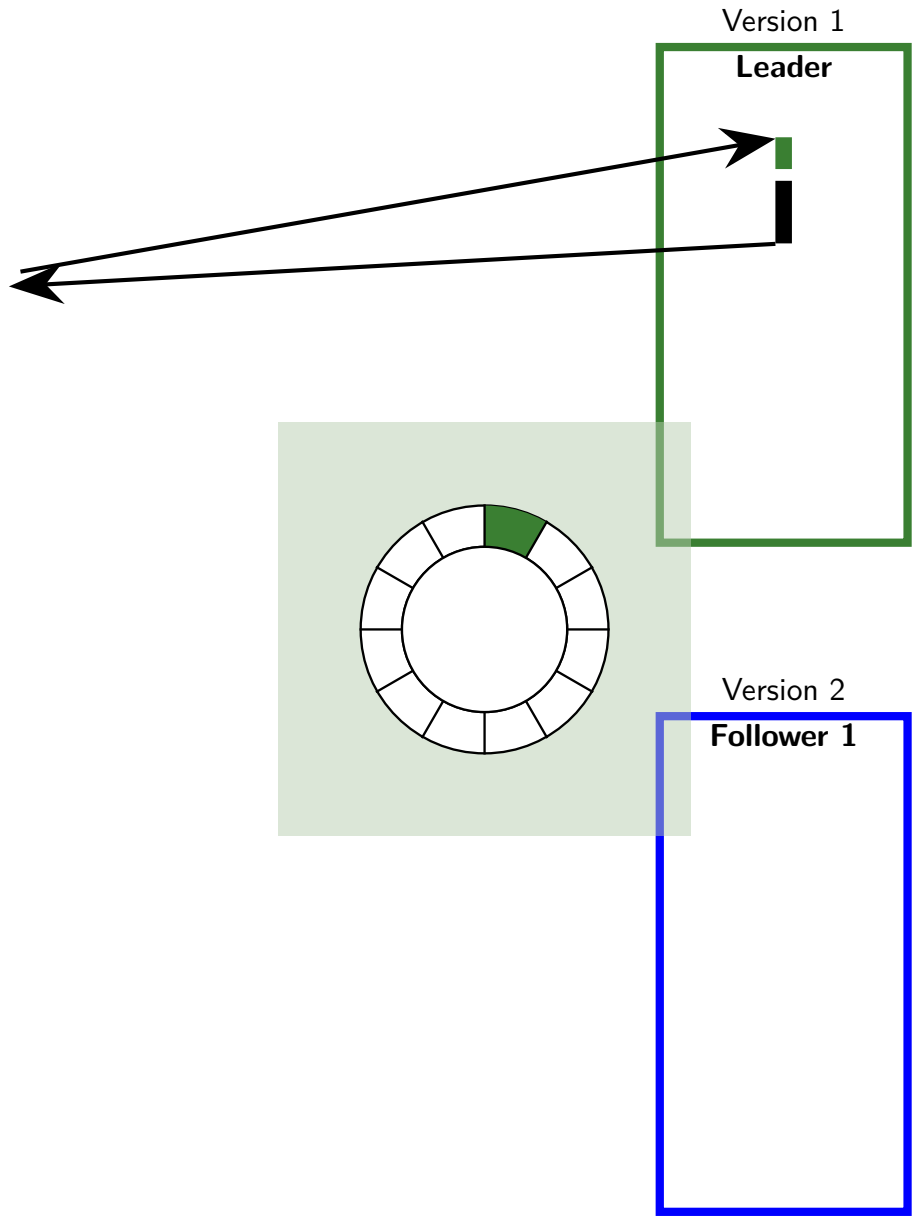
Varan



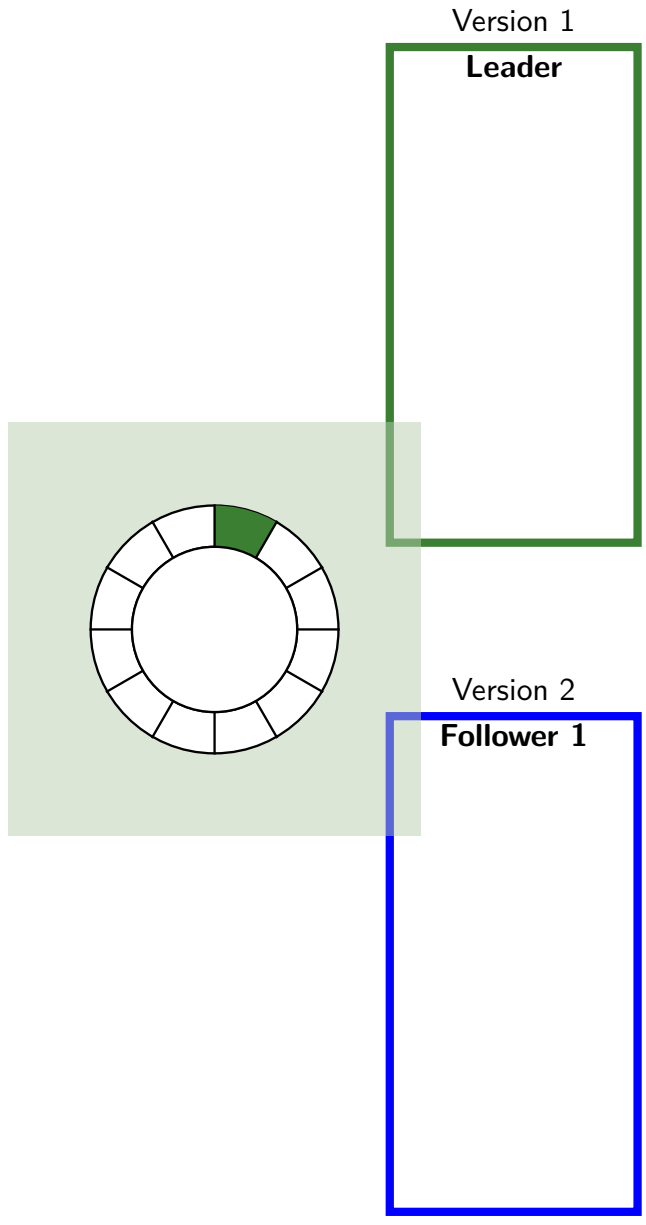
Varan



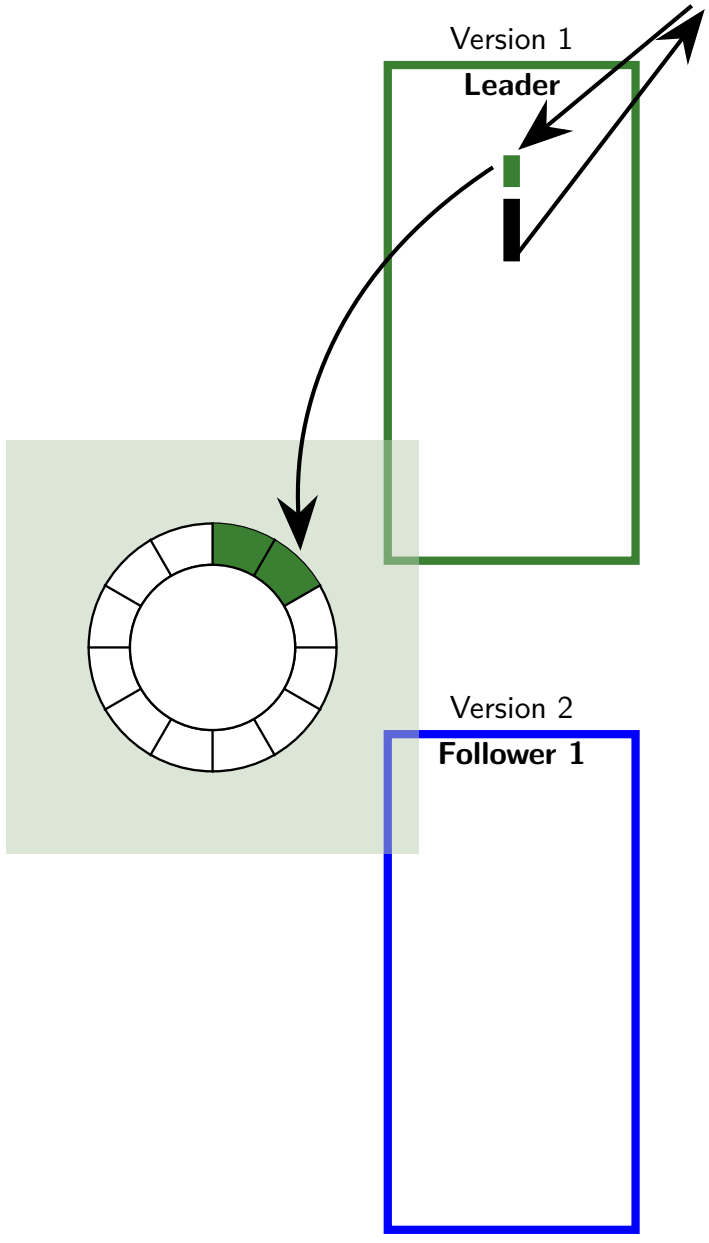
Varan



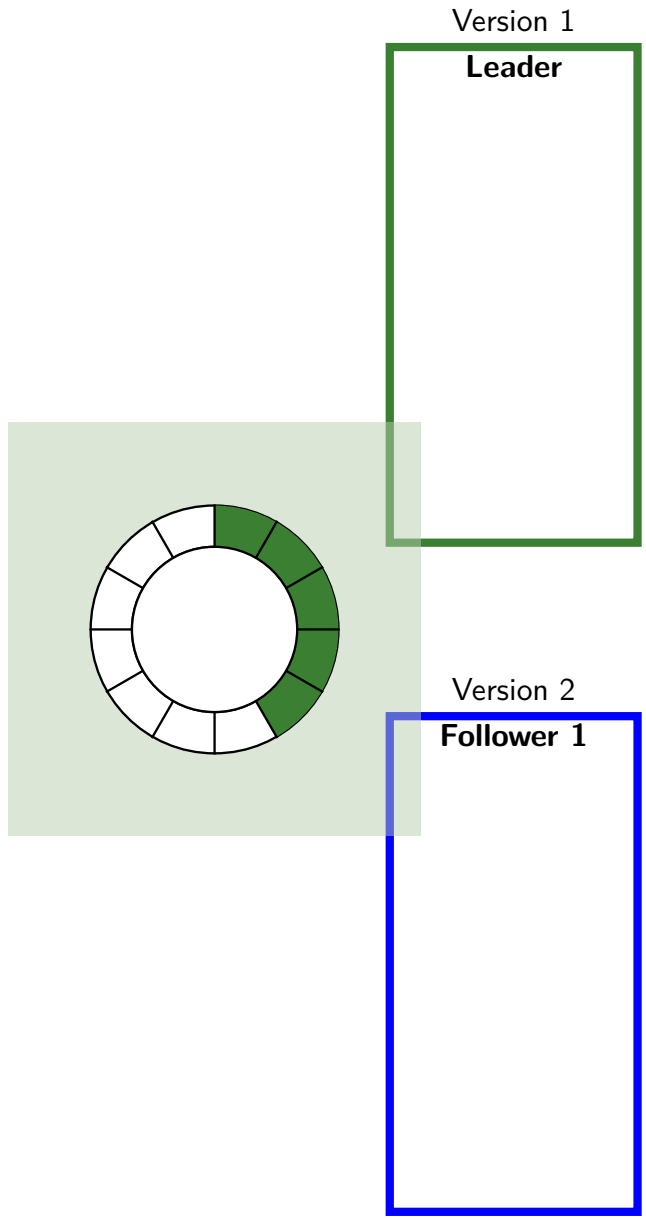
Varan



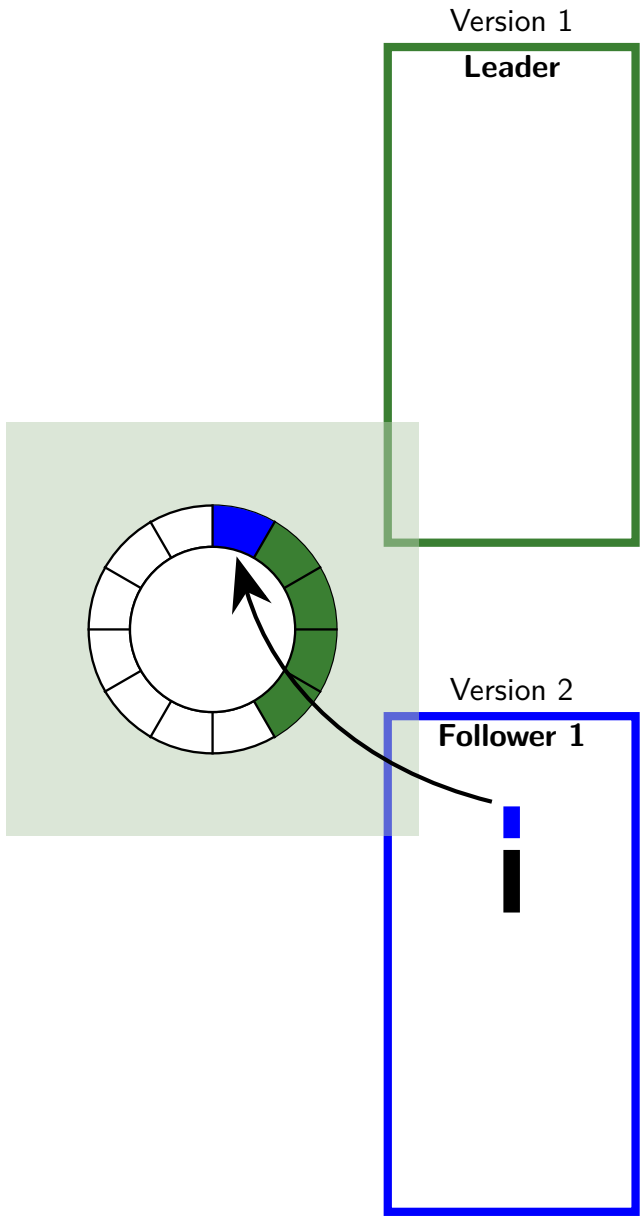
Varan



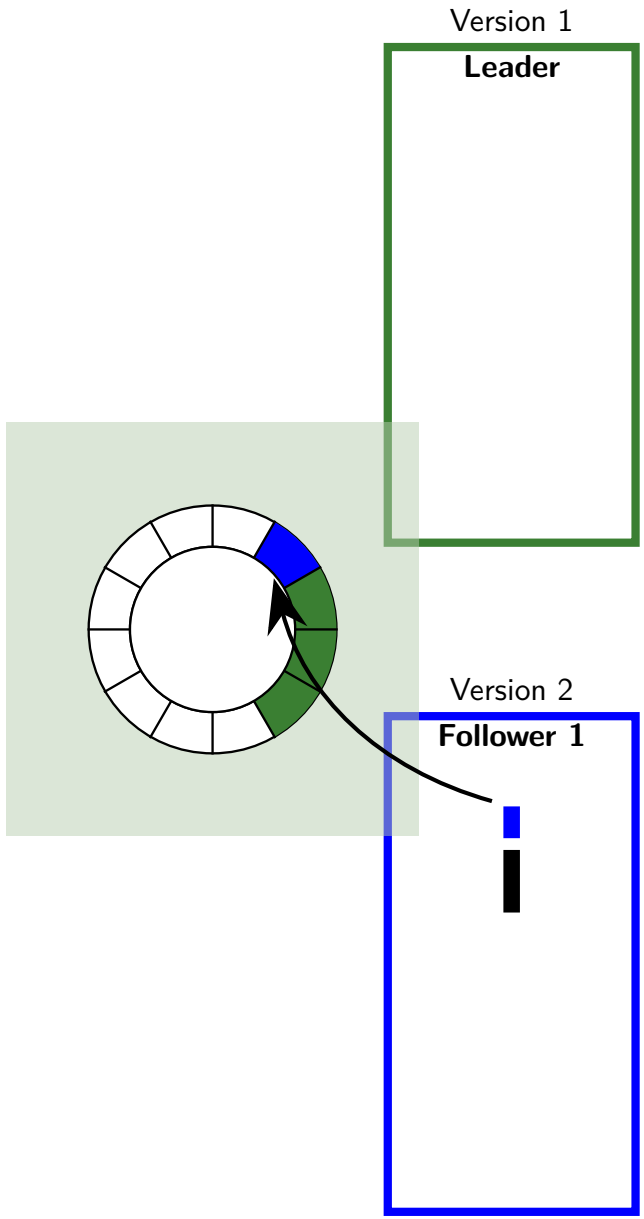
Varan



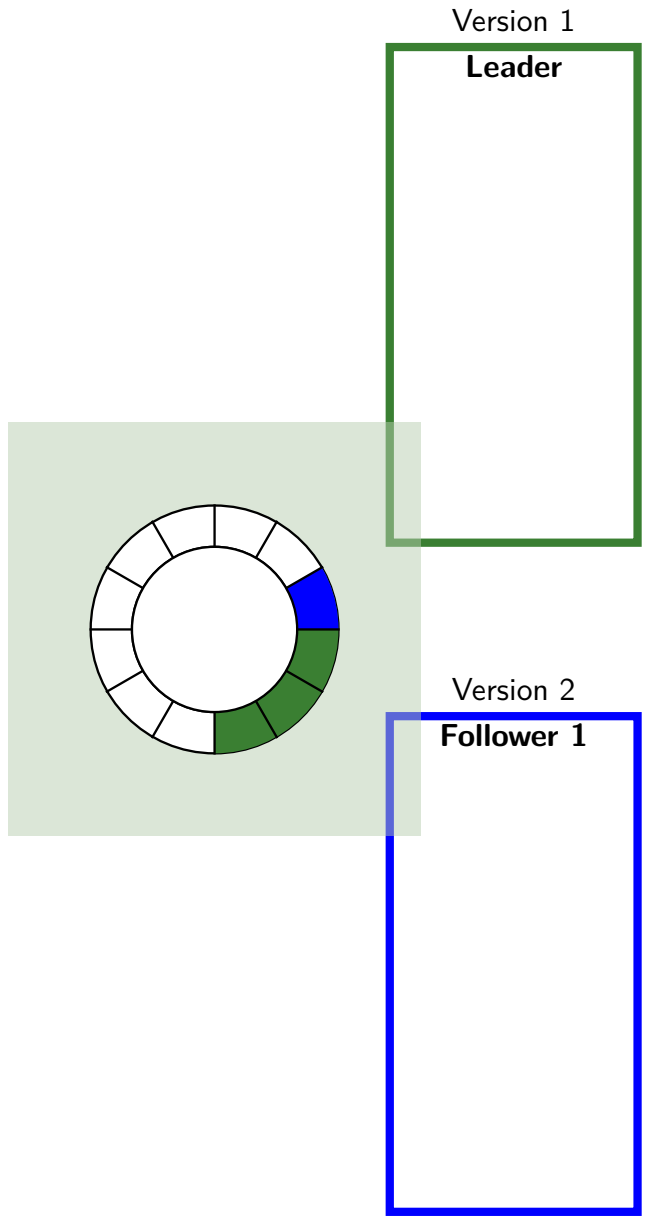
Varan



Varan



Varan



Varan

System calls

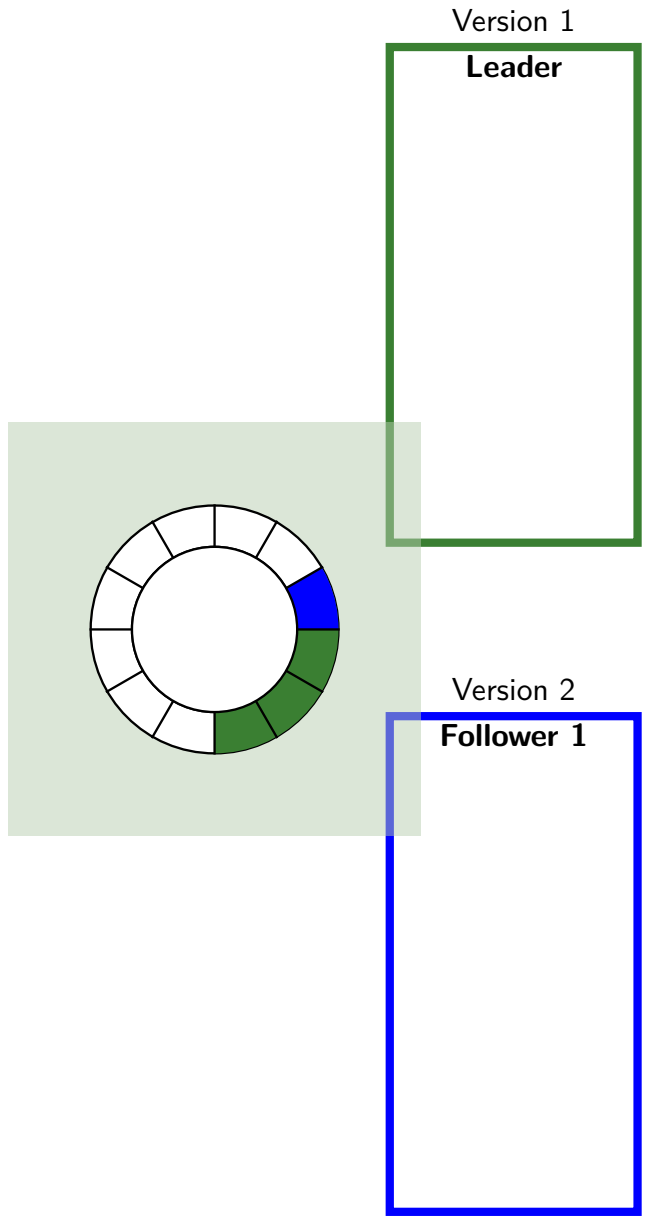
```
01
02      while (true) {
03
04          sckt = accept();          // Wait for client
05
06          req = parse(read(skt)); // Handle request
07          file = open(req);
08          rsp = read(file);
09
10
11
12          write(skt, rsp);          // Send response
13
14      }
15
```

Varan

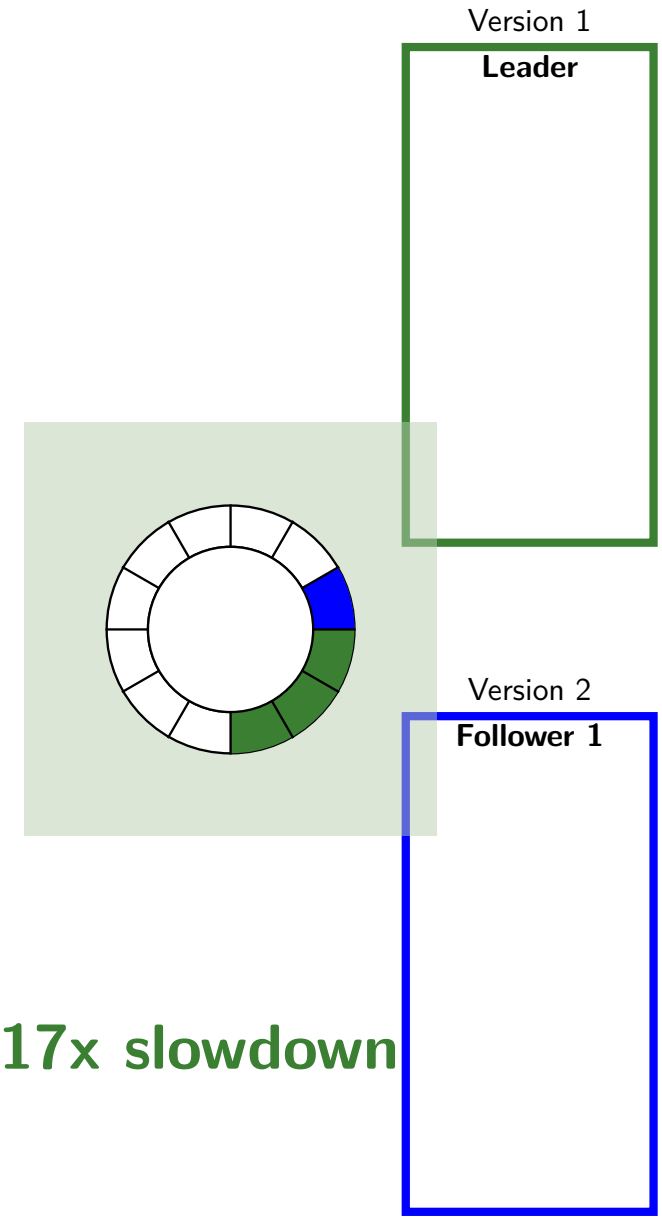
System calls

```
01
02      while (true) {
03
04          sckt = accept();            // Wait for client
05
06          req = parse(read(skt));     // Handle request
07          file = open(req);
08          rsp = read(file);
09
10
11
12          write(skt, rsp);            // Send response
13
14      }
15
```

Varan

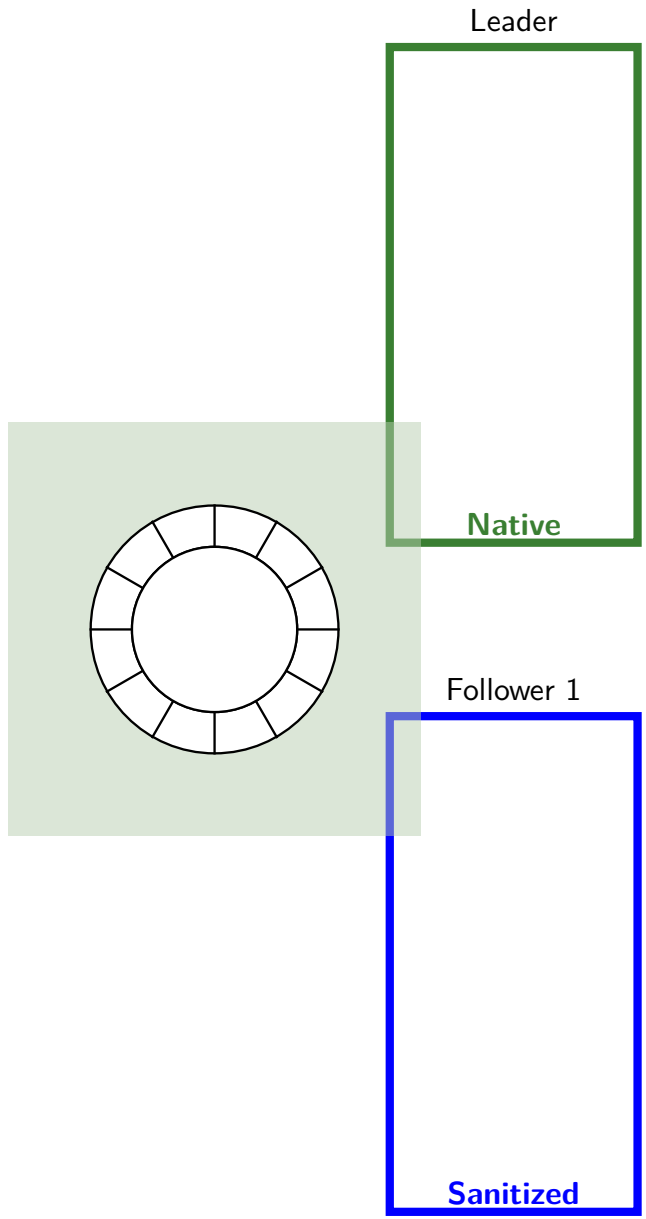


Varan

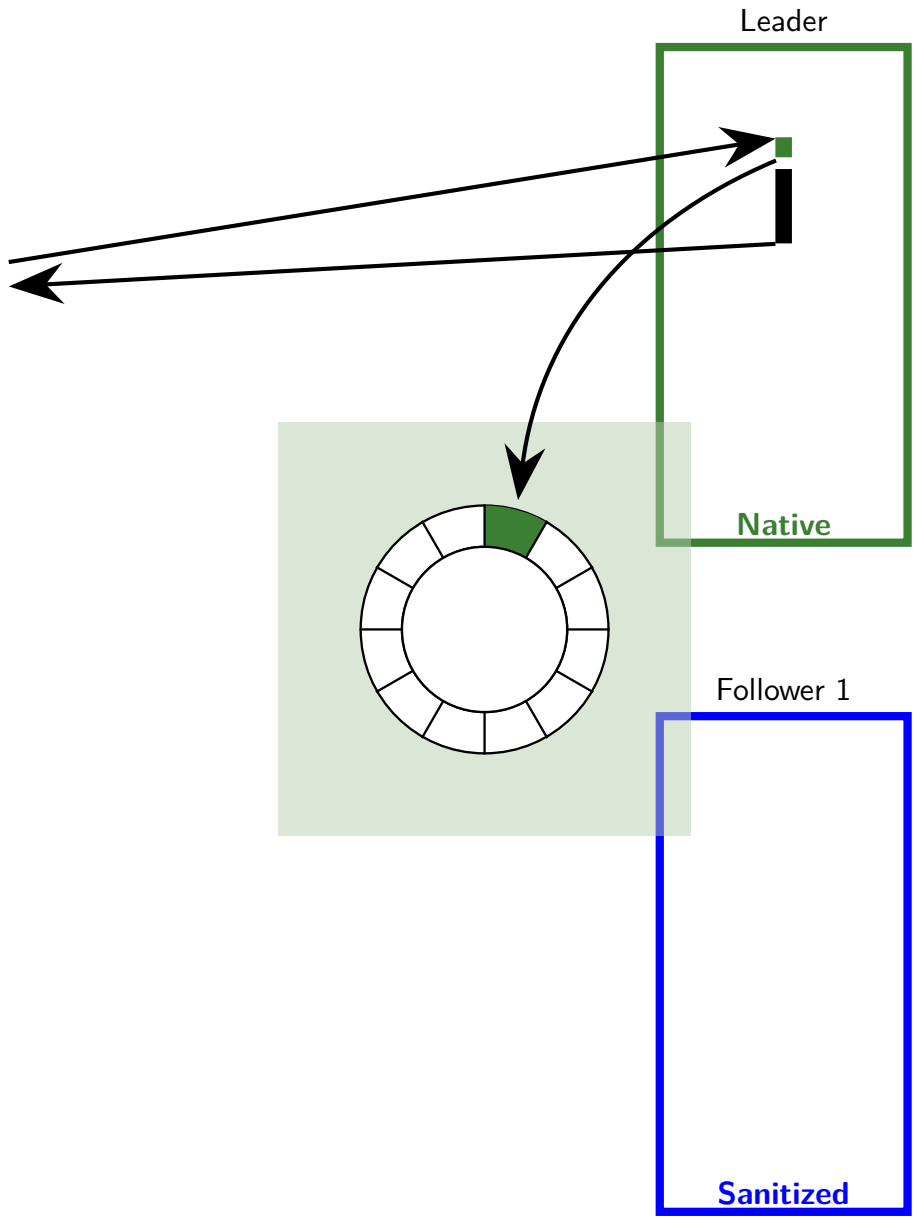


1x–1.17x slowdown

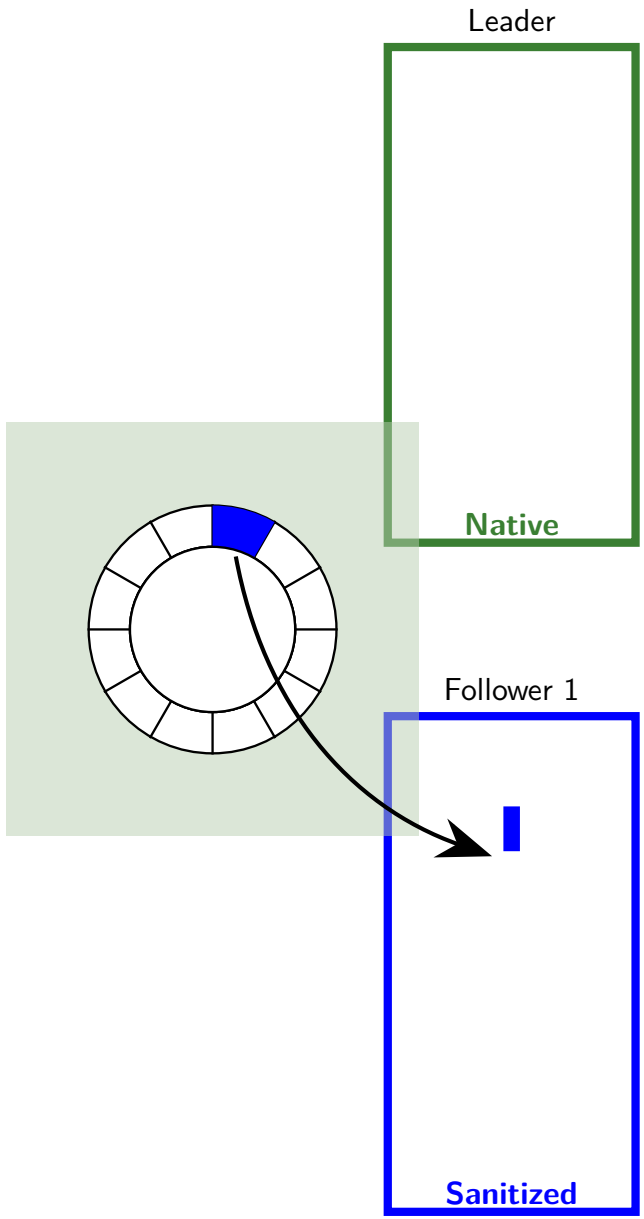
Varan + Dynamic Analysis



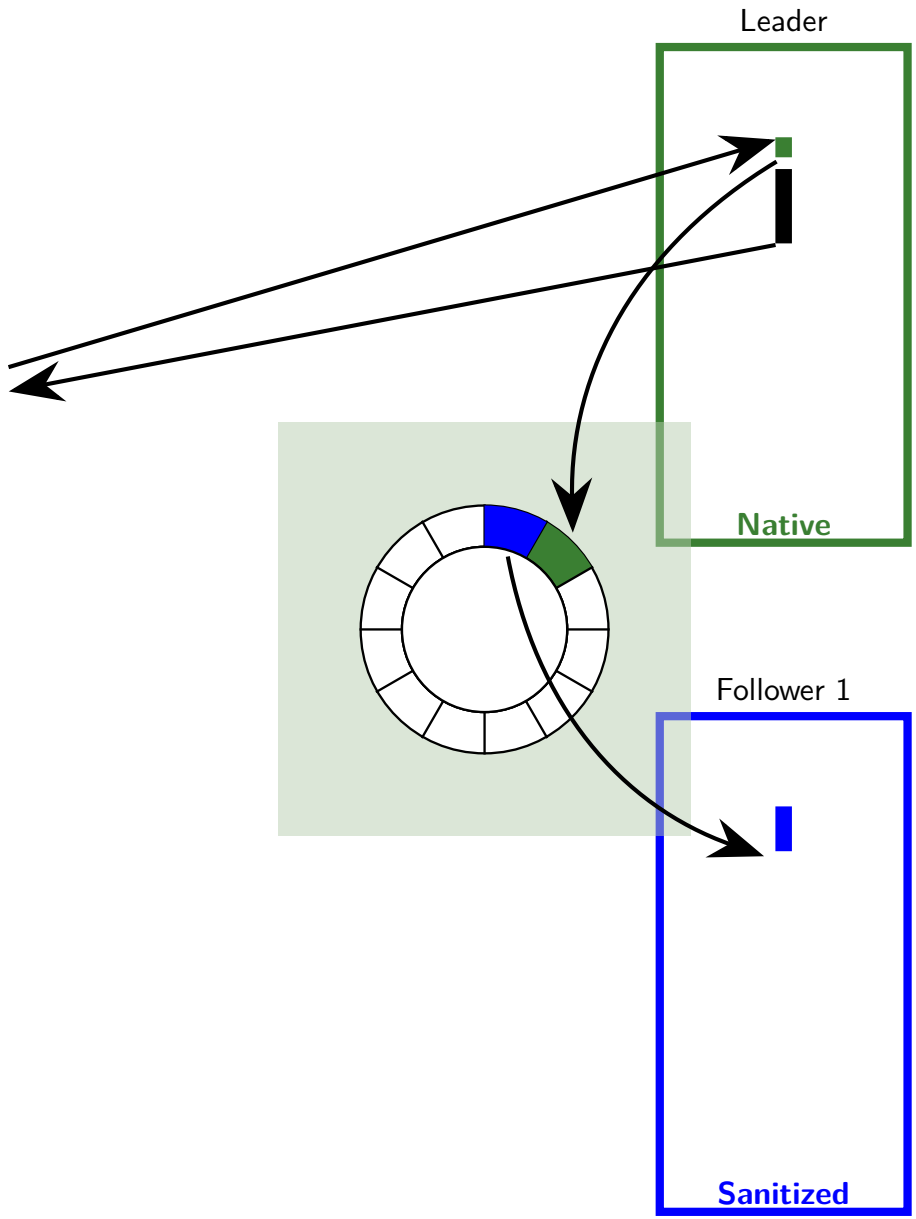
Varan + Dynamic Analysis



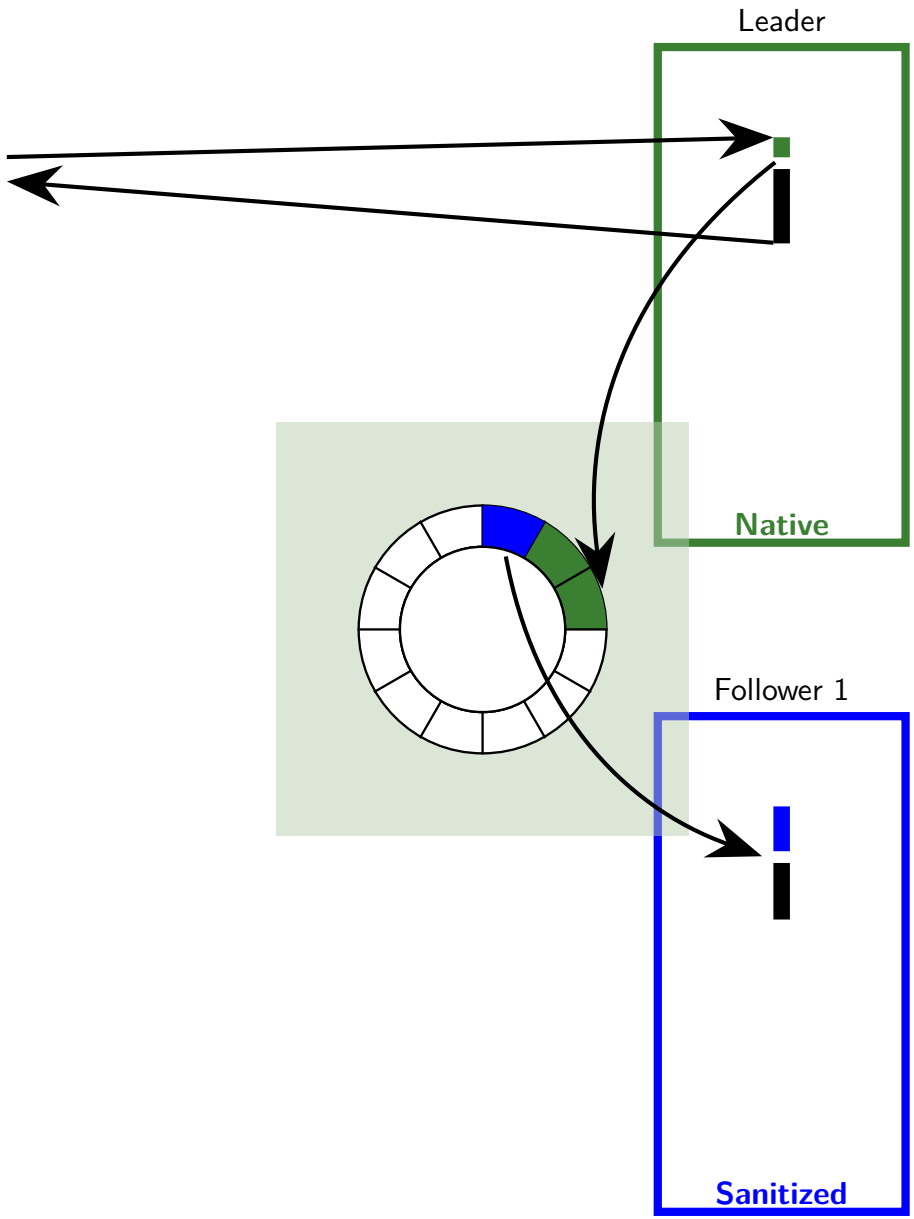
Varan + Dynamic Analysis



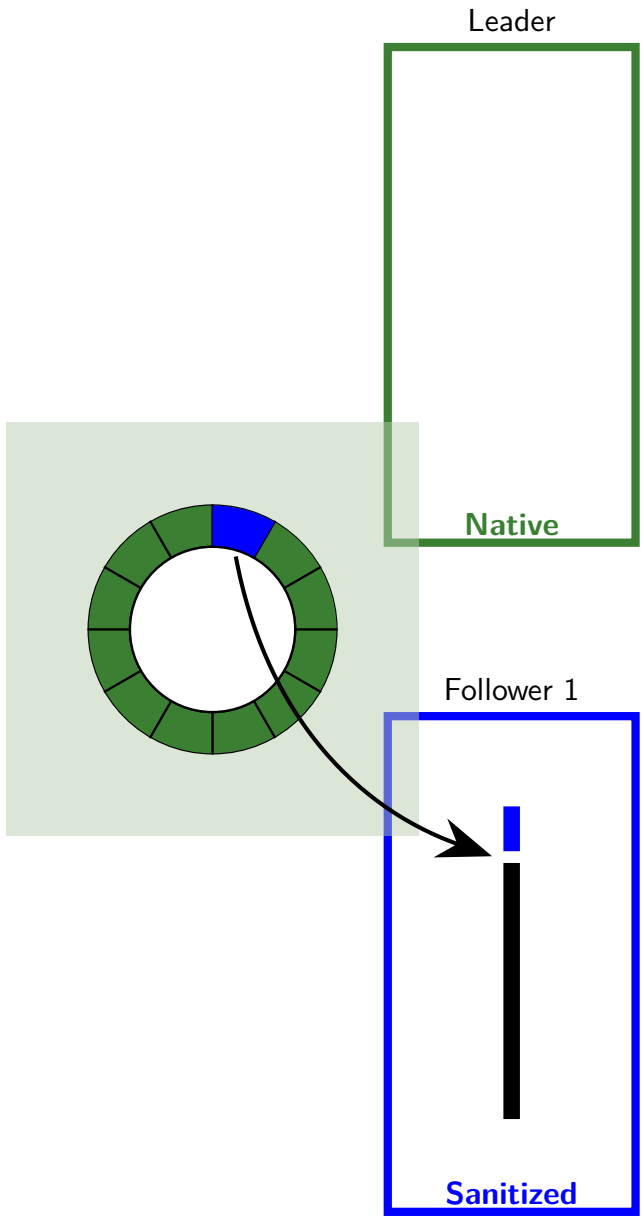
Varan + Dynamic Analysis



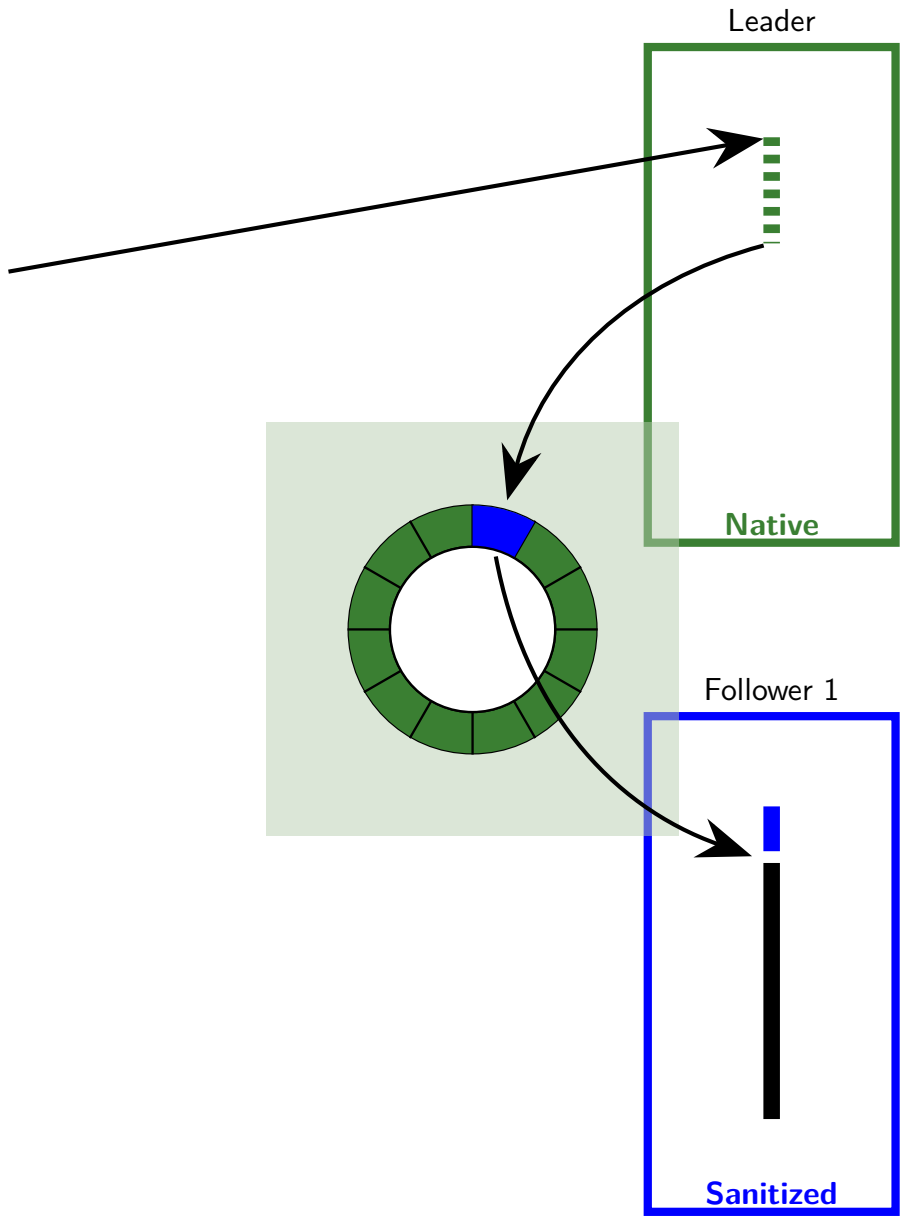
Varan + Dynamic Analysis



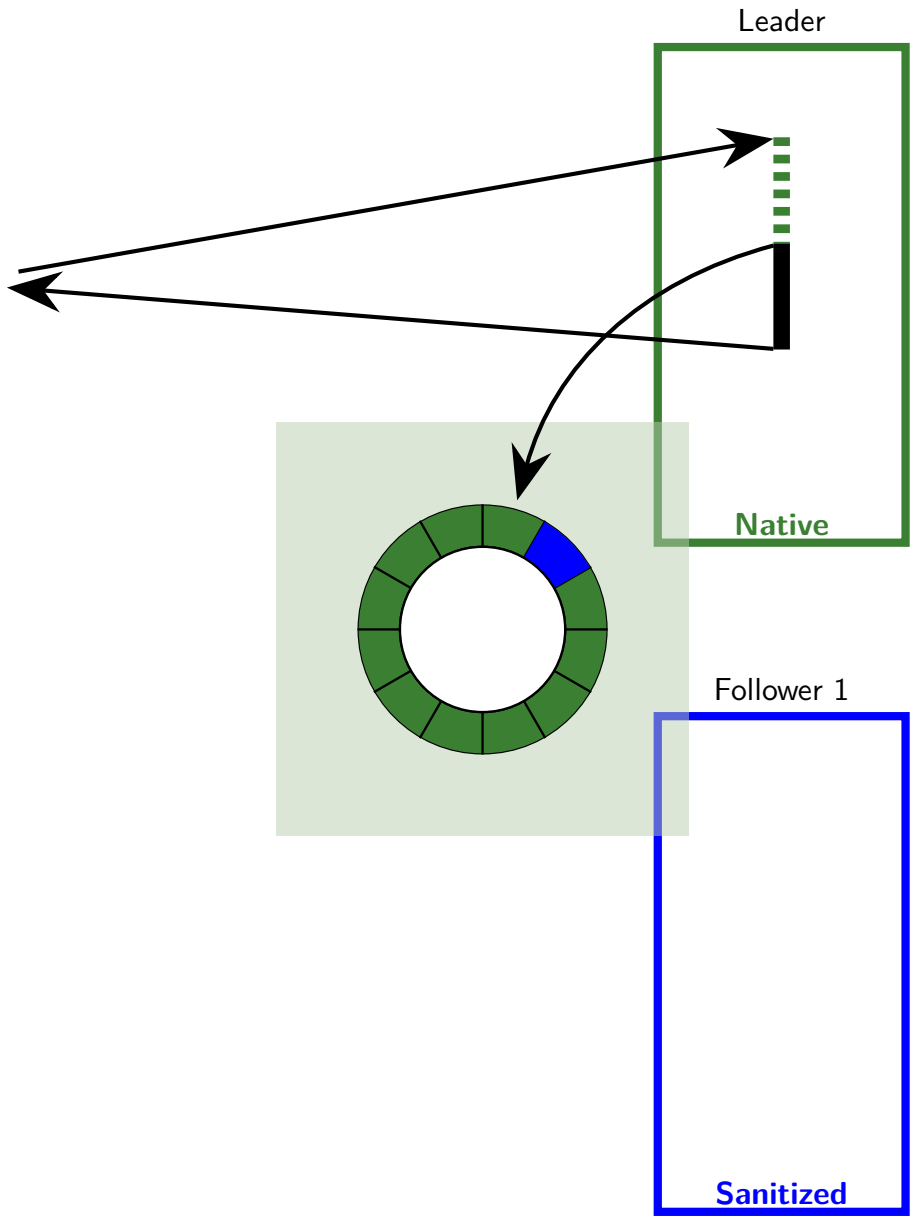
Varan + Dynamic Analysis



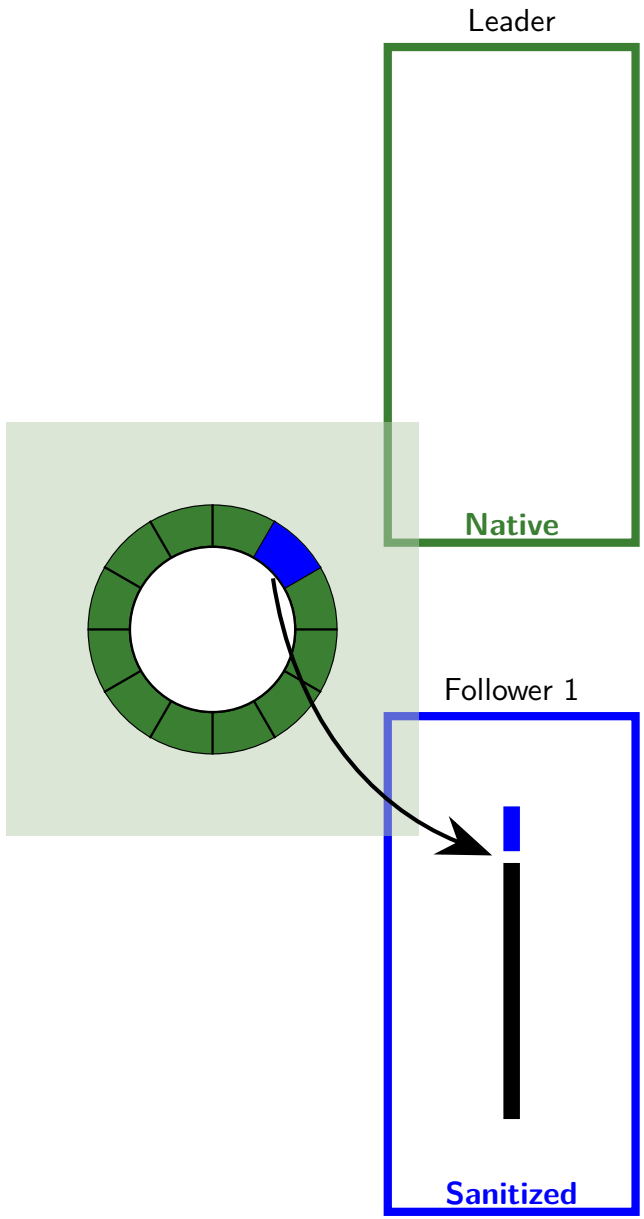
Varan + Dynamic Analysis



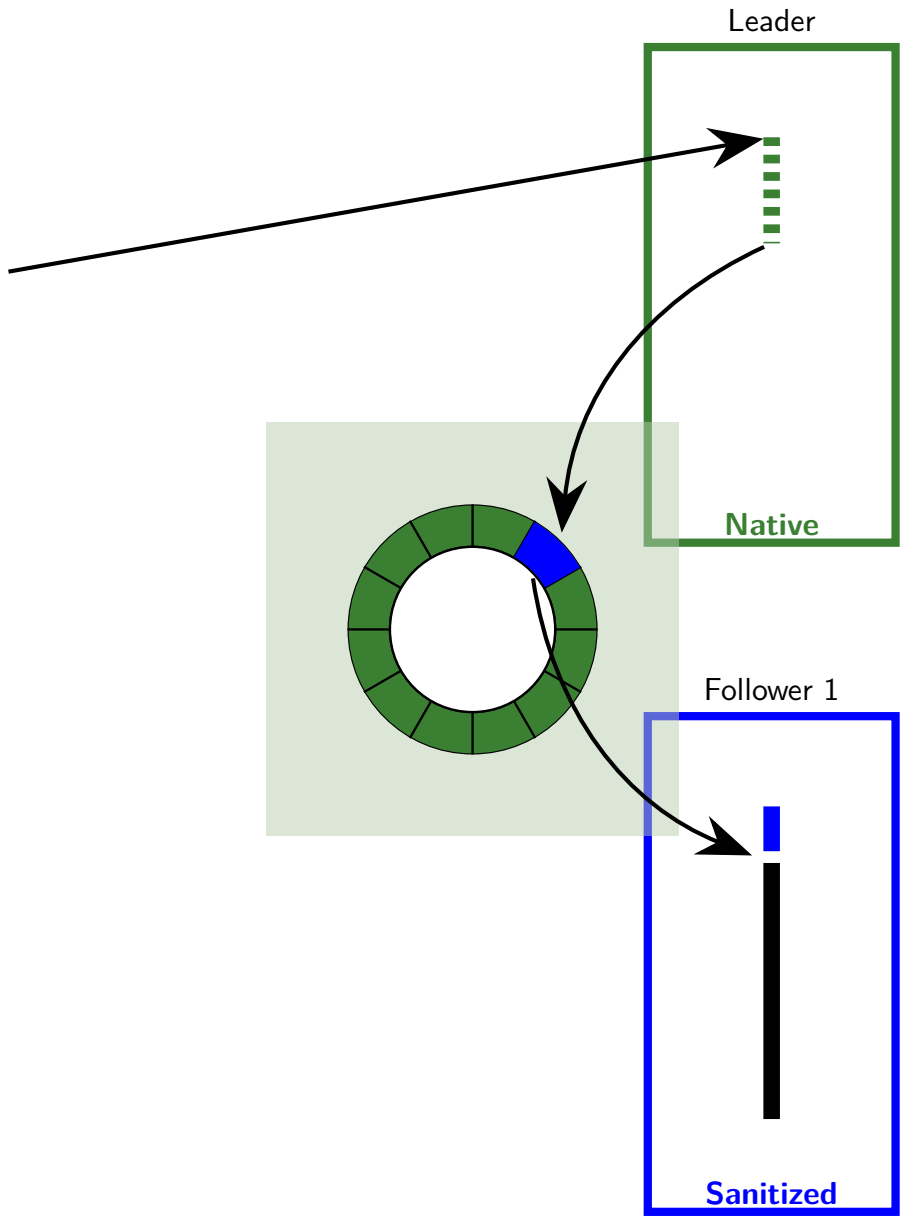
Varan + Dynamic Analysis



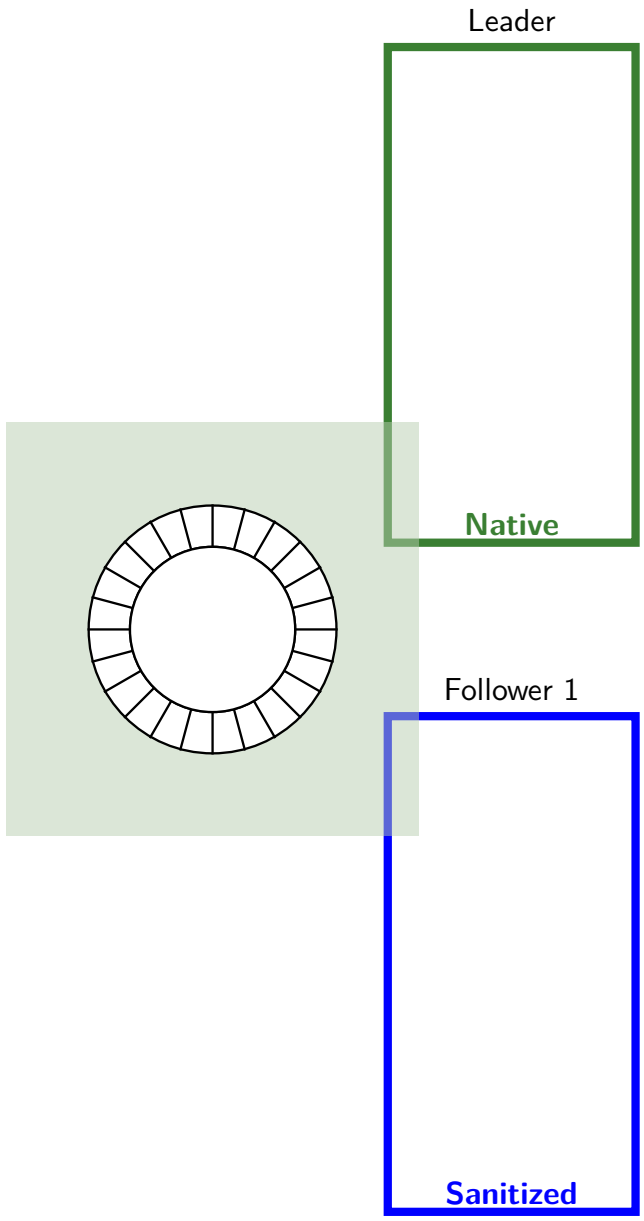
Varan + Dynamic Analysis



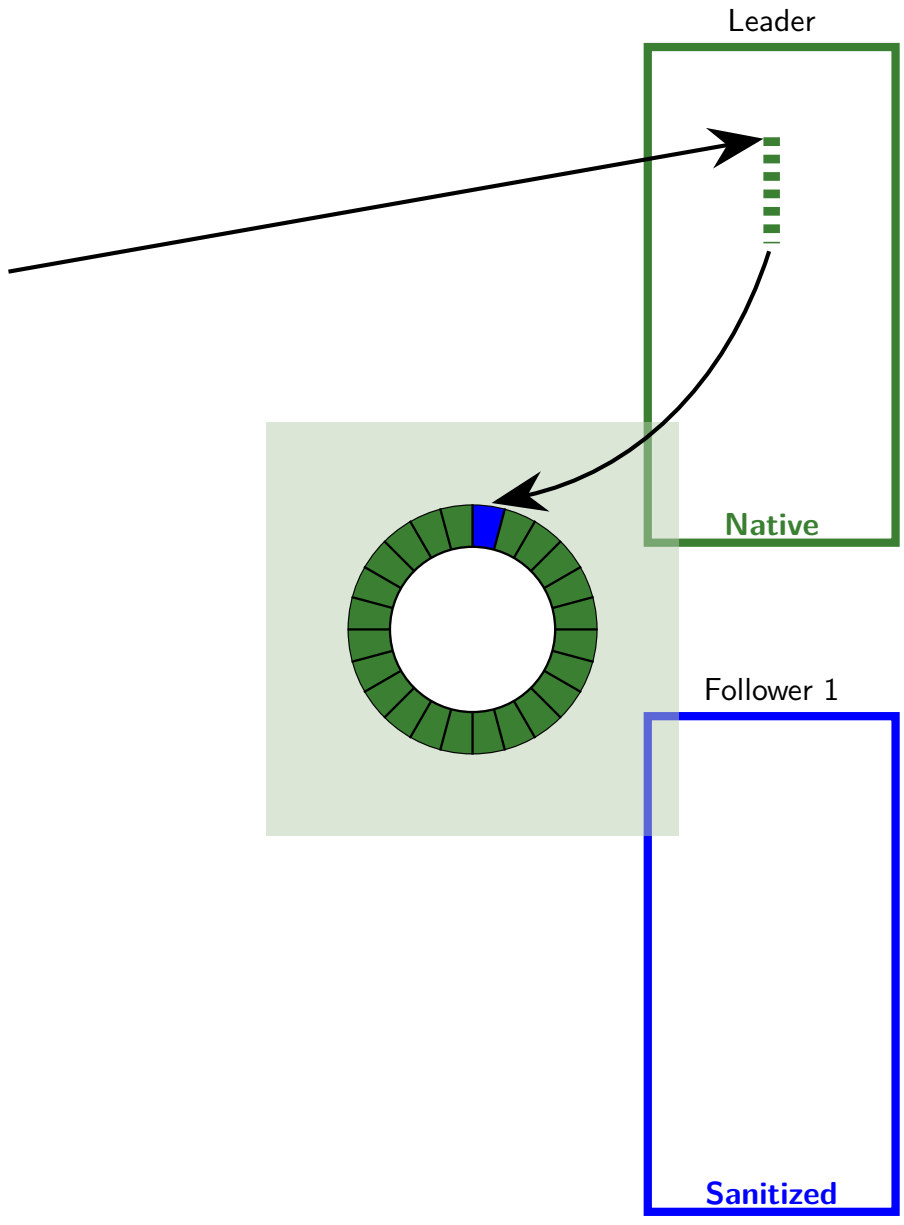
Varan + Dynamic Analysis



Larger ringbuffer?



Larger ringbuffer?

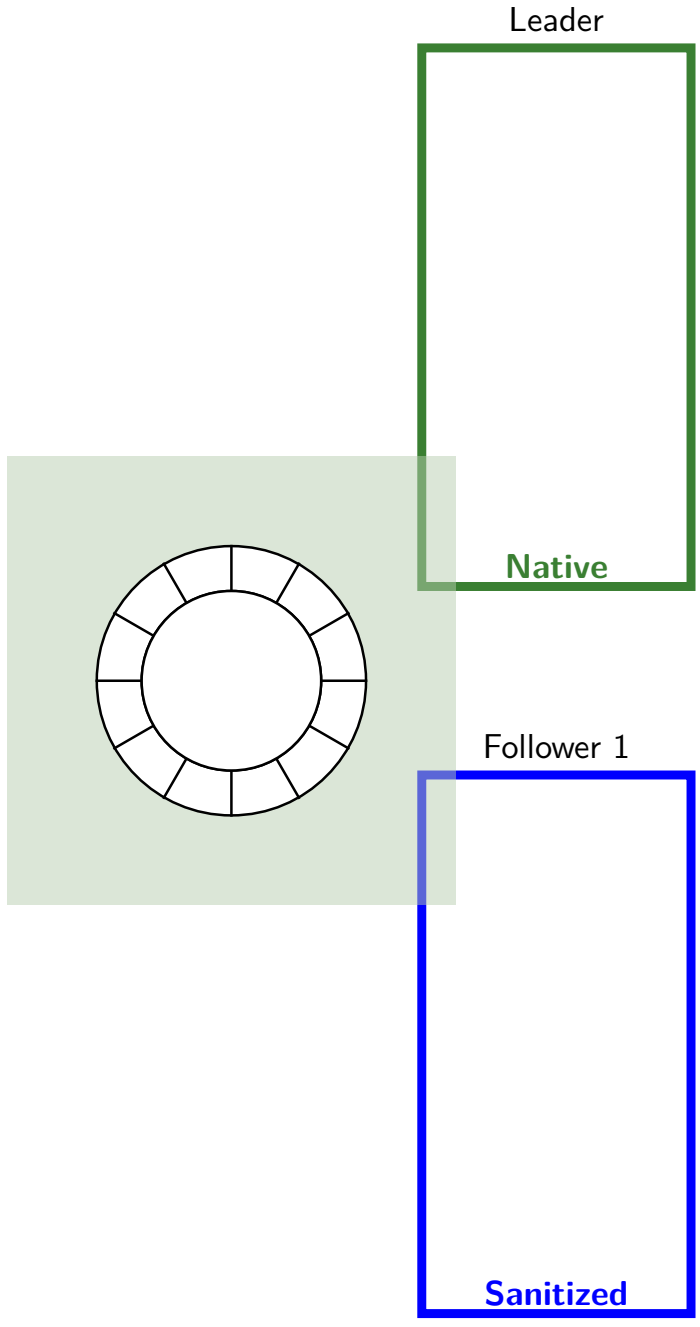


Larger ringbuffer

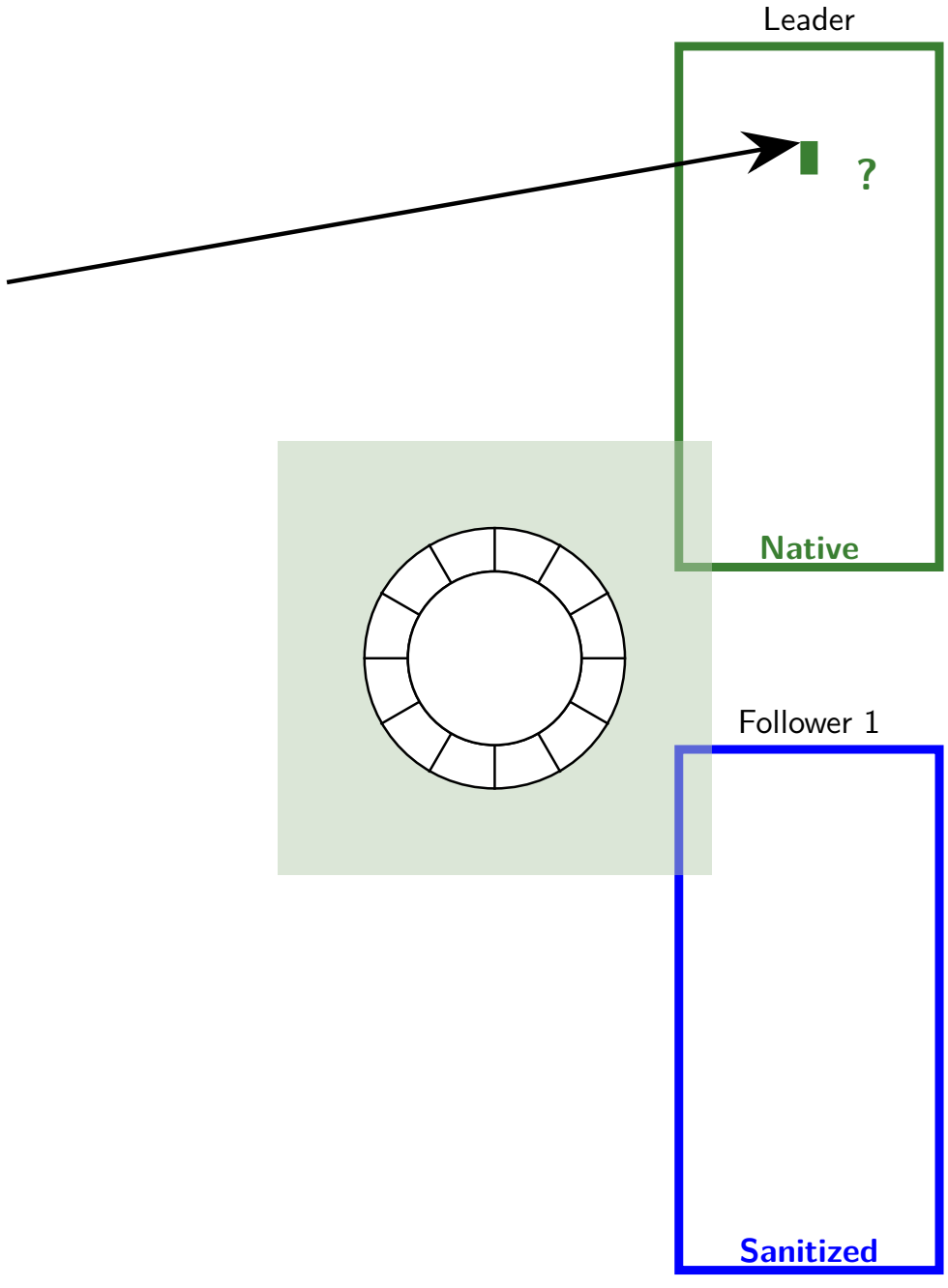
Interactive applications

- `alias vim='vx vim vim-asan'`
- `alias htop='vx htop htop-asan'`
- `alias mutt='vx mutt mutt-asan'`
- `alias ssh='vx ssh ssh-asan'`
- `alias ls='vx ls ls-asan'`

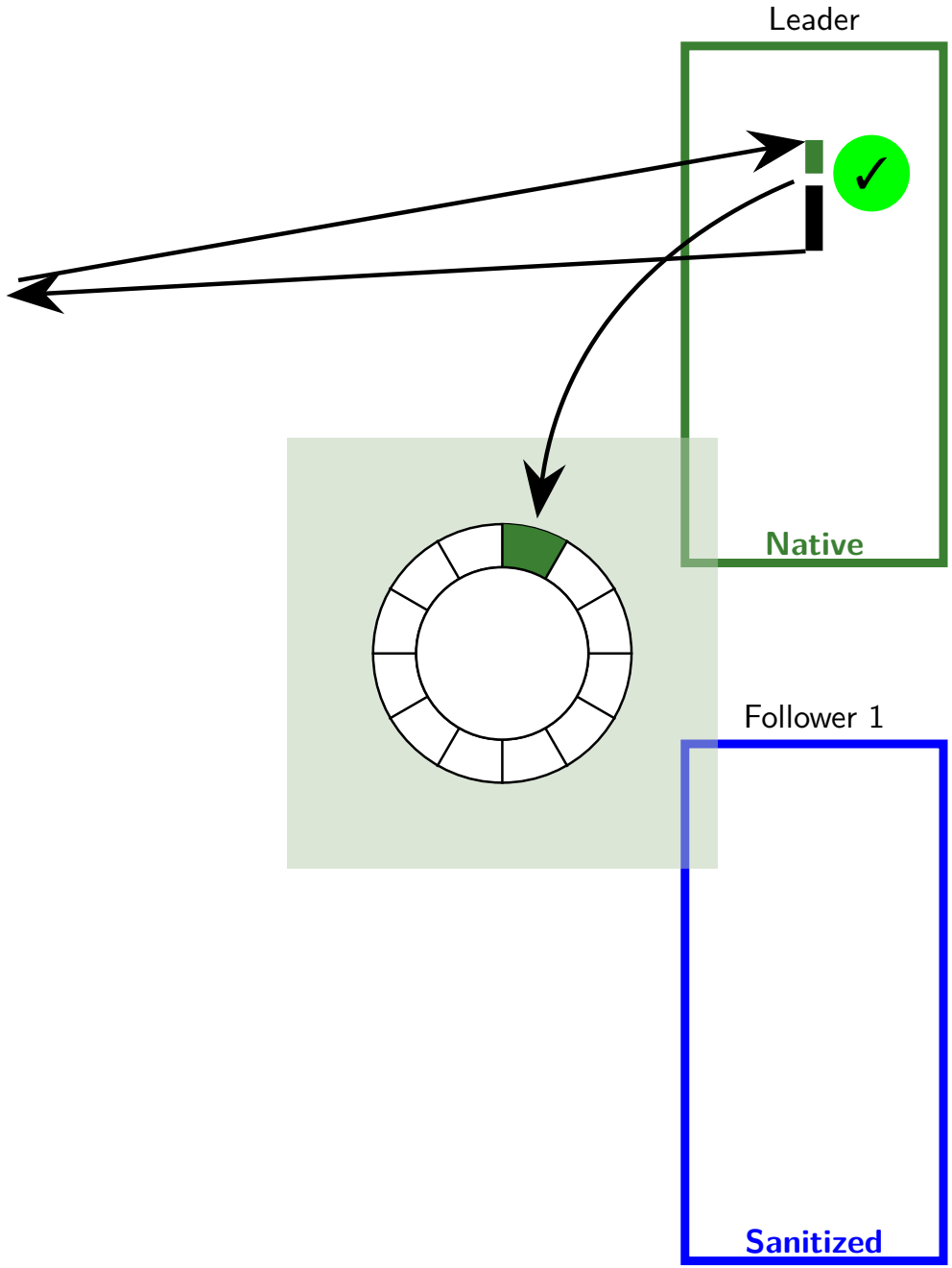
Drop requests



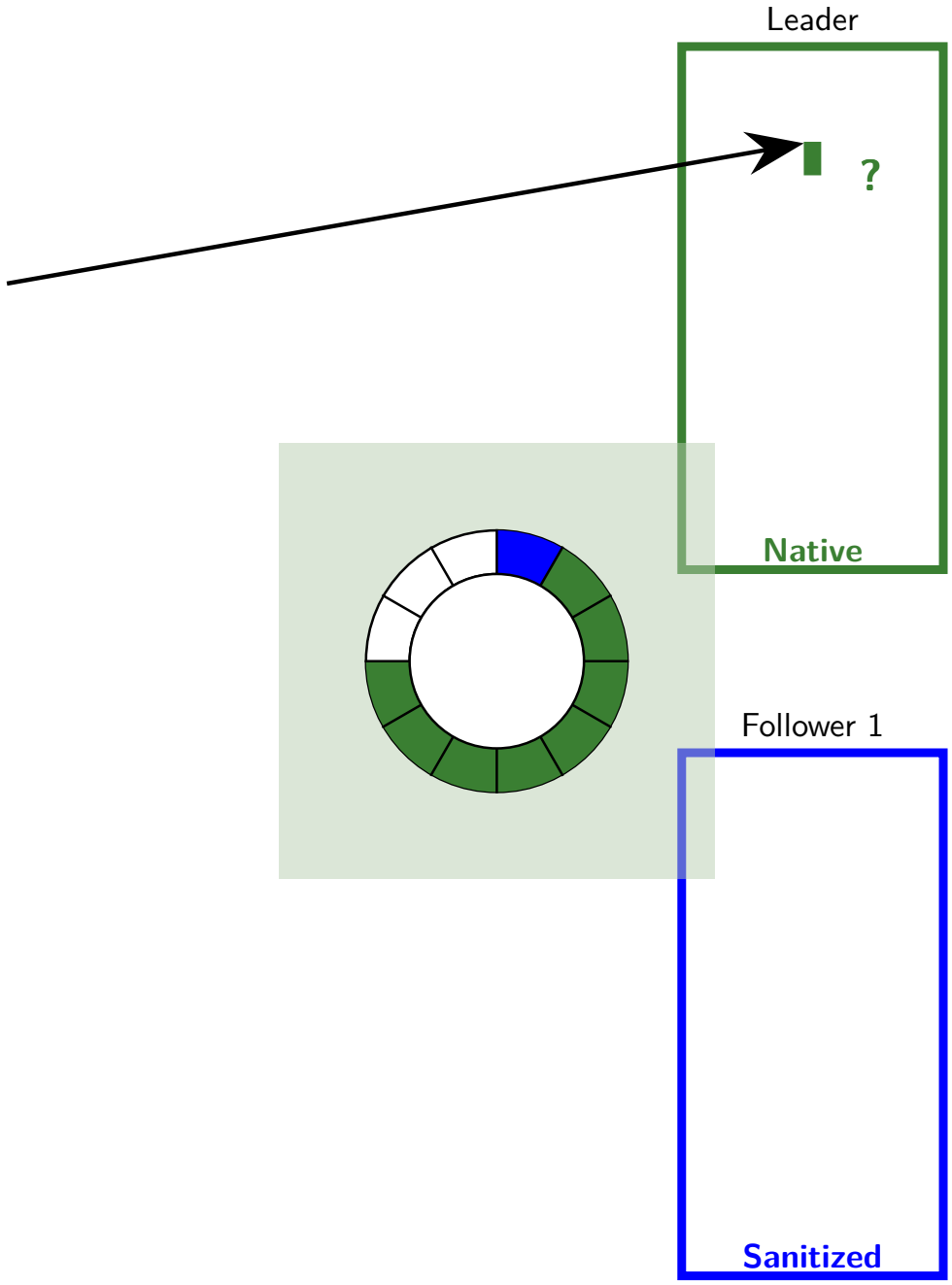
Drop requests



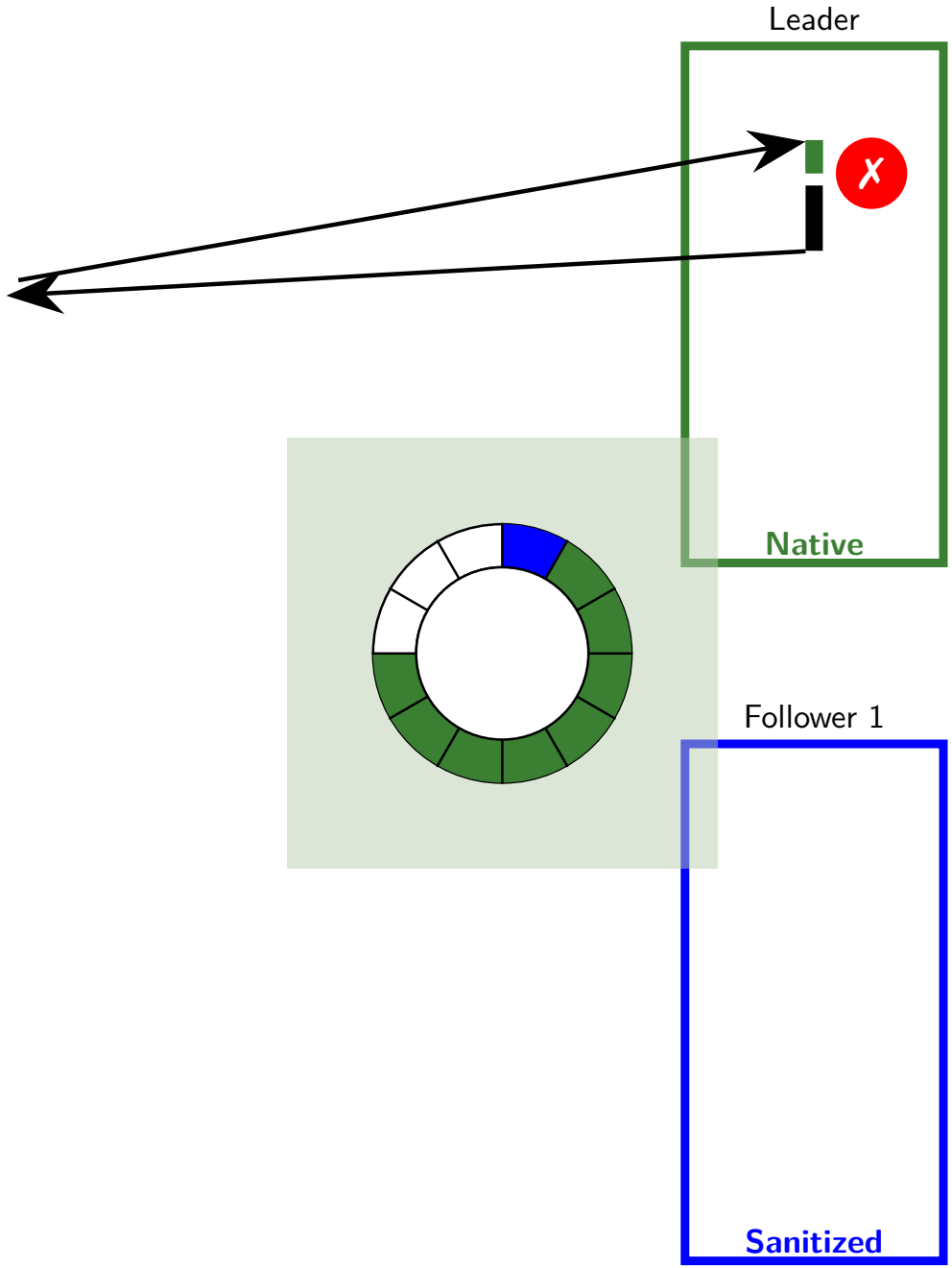
Drop requests



Drop requests



Drop requests

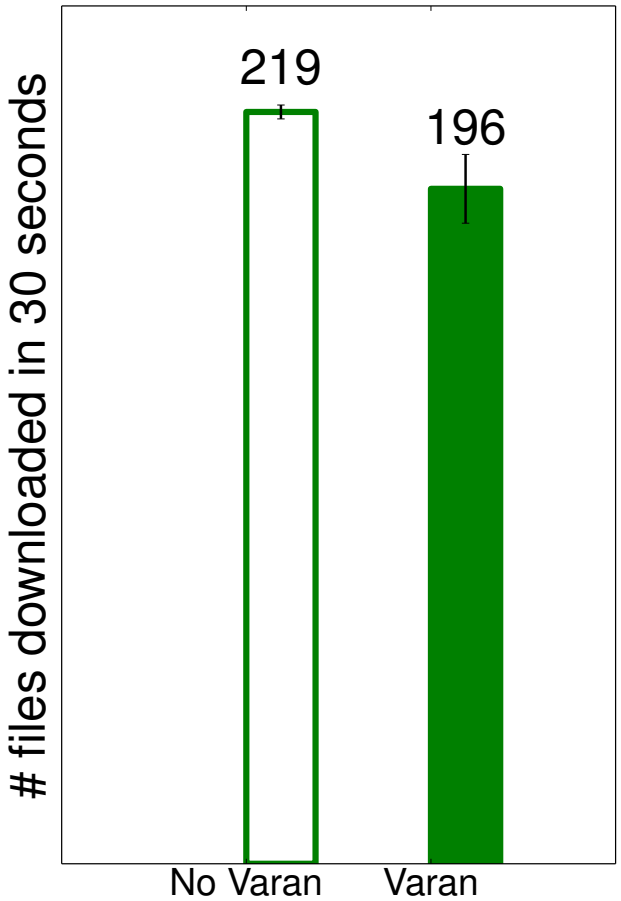


Experiment

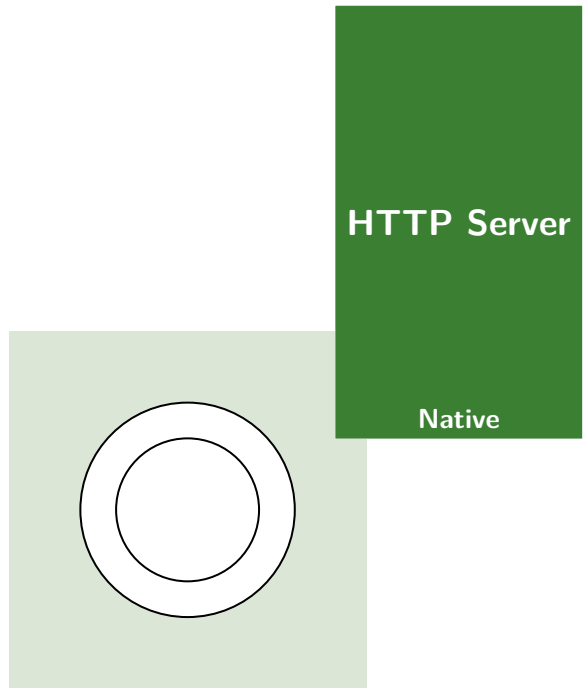
Check performance with varying drop rate on simple HTTP server

- Single process/thread
- `gcc -fsanitize=address` on follower
- BZip2 the response

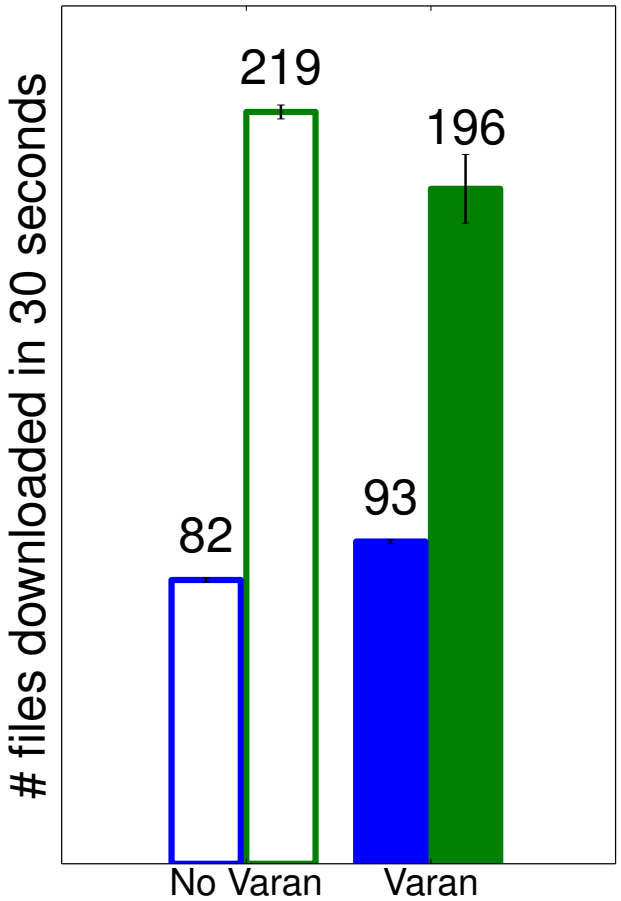
Results


219
196
files downloaded in 30 seconds
No Varan
Varan

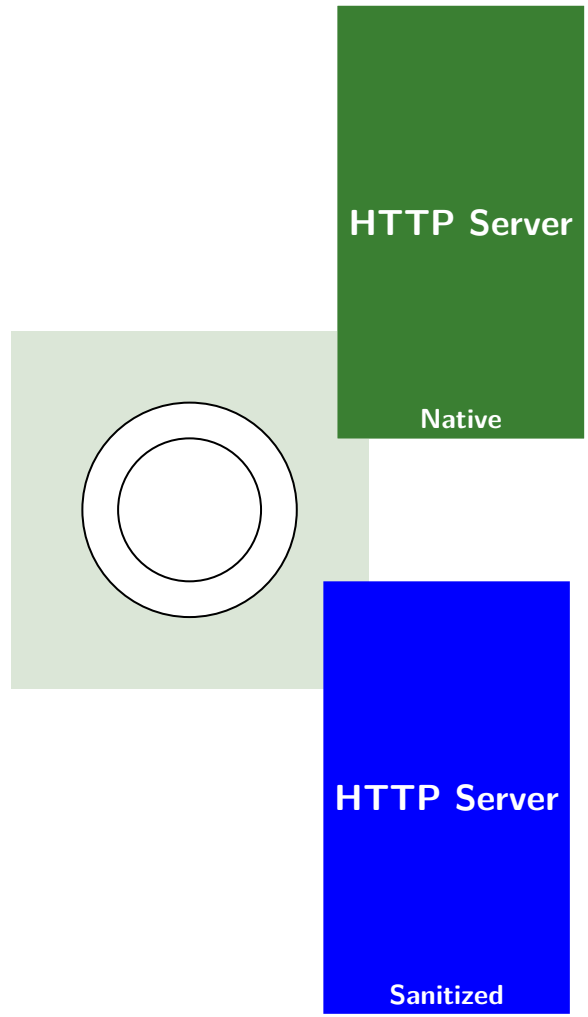

HTTP Server
Native


HTTP Server
Native

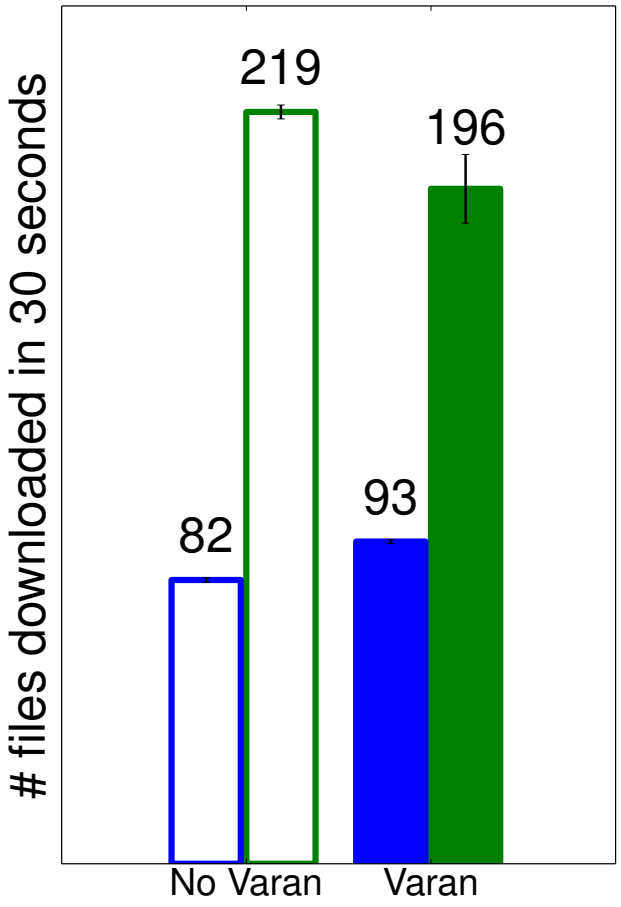
Results

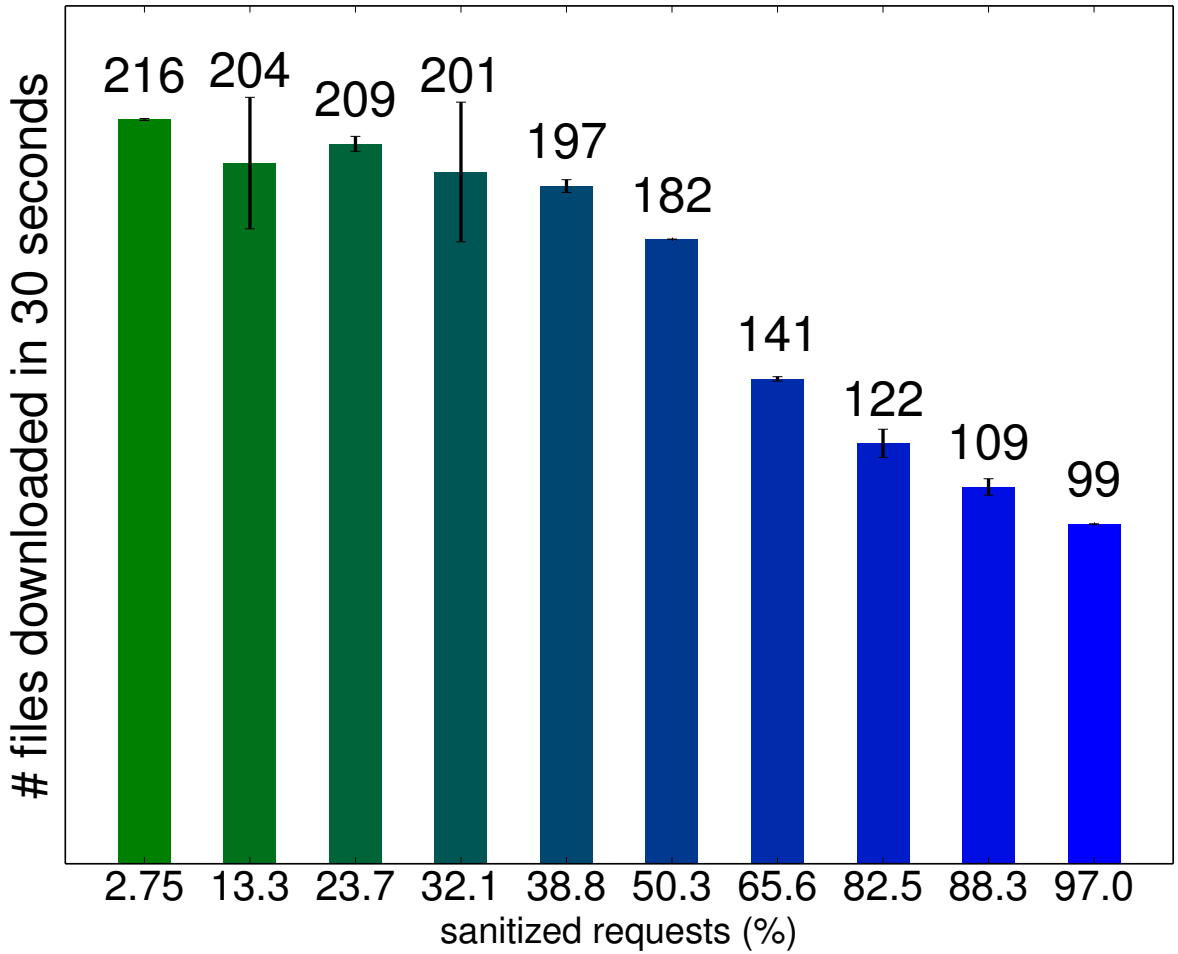

files downloaded in 30 seconds
219
196
93
82
No Varan
Varan


HTTP Server
Sanitized


HTTP Server
Native
HTTP Server
Sanitized

Results


files downloaded in 30 seconds
219
196
82
93
No Varan
Varan


files downloaded in 30 seconds
216
204
209
201
197
182
141
122
109
99
2.75
13.3
23.7
32.1
38.8
50.3
65.6
82.5
88.3
97.0
sanitized requests (%)

Multi-version execution for security

a or b?

```
01 int x = 1;
02
03 void f(void) {
04   if (5 % (3 * x) + 2 != 4)
05     puts("a");
06   else
07     puts("b");
08 }
```

a or b?

```
01 int x = 1;
02
03 void f(void) {
04   if (5 % (3 * x) + 2 != 4)
05     puts("a");
06   else
07     puts("b");
08 }
```

Clang 3.3 says **a**

https://llvm.org/bugs/show_bug.cgi?id=15940

sudo

```
01 htop
02 # command not found: htop
03
04 apt-get install htop
05 # Permission denied, are you root?
06
07 sudo apt-get install htop
08 # Installing...
09
10 htop
11 # running htop...
```

sudo

```
01 whoami
02 # user
03
04 sudo whoami
05 # root
```

sudo backdoor

```
01 gcc sudo.c -o sudo-gcc
02 clang sudo.c -o sudo-clang
03
04 ./sudo-gcc whomai
05 # user is not in the sudoers file.
06 # This incident will be reported.
07
08 ./sudo-clang whomai
09 # root
10
11
12
```

https://github.com/regehr/sudo-1.8.13/tree/compromise/backdoor-info

sudo backdoor

```
01 gcc sudo.c -o sudo-gcc
02 clang sudo.c -o sudo-clang
03
04 ./sudo-gcc whomai
05 # user is not in the sudoers file.
06 # This incident will be reported.
07
08 ./sudo-clang whomai
09 # root
10
11 vx ./sudo-gcc ./sudo-clang -- whoami
12 # divergence detected, terminating
```

Cristian Cadar and Luís Pina and John Regehr, *Multi-Version Execution Defeats a Compiler-Bug-Based Backdoor*, http://blog.regehr.org/archives/1282, 2015

Undefined behavior

```
01 int saturating_add(int x, int y) {
02   if(x > 0 && y > 0 && x + y < 0)
03     return INT_MAX;
04   if(x < 0 && y < 0 && x + y > 0)
05     return INT_MIN;
06   return x + y;
07 }
```

Undefined behavior

```
01 int saturating_add(int x, int y) {
02   if(x > 0 && y > 0 && x + y < 0)
03     return INT_MAX;
04   if(x < 0 && y < 0 && x + y > 0)
05     return INT_MIN;
06   return x + y;
07 }
```

x	y	Result
1	2	3

Undefined behavior

```
01 int saturating_add(int x, int y) {
02   if(x > 0 && y > 0 && x + y < 0)
03     return INT_MAX;
04   if(x < 0 && y < 0 && x + y > 0)
05     return INT_MIN;
06   return x + y;
07 }
```

x	y	Result
1	2	3
1000000000	1000000000	2000000000

Undefined behavior

```
01 int saturating_add(int x, int y) {
02   if(x > 0 && y > 0 && x + y < 0)
03     return INT_MAX;
04   if(x < 0 && y < 0 && x + y > 0)
05     return INT_MIN;
06   return x + y;
07 }
```

x	y	Result
1	2	3
1000000000	1000000000	2000000000
2000000000	2000000000	2147483647

Undefined behavior

```
01 int saturating_add(int x, int y) {
02   if(x > 0 && y > 0 && x + y < 0)
03     return INT_MAX;
04   if(x < 0 && y < 0 && x + y > 0)
05     return INT_MIN;
06   return x + y;
07 }
```

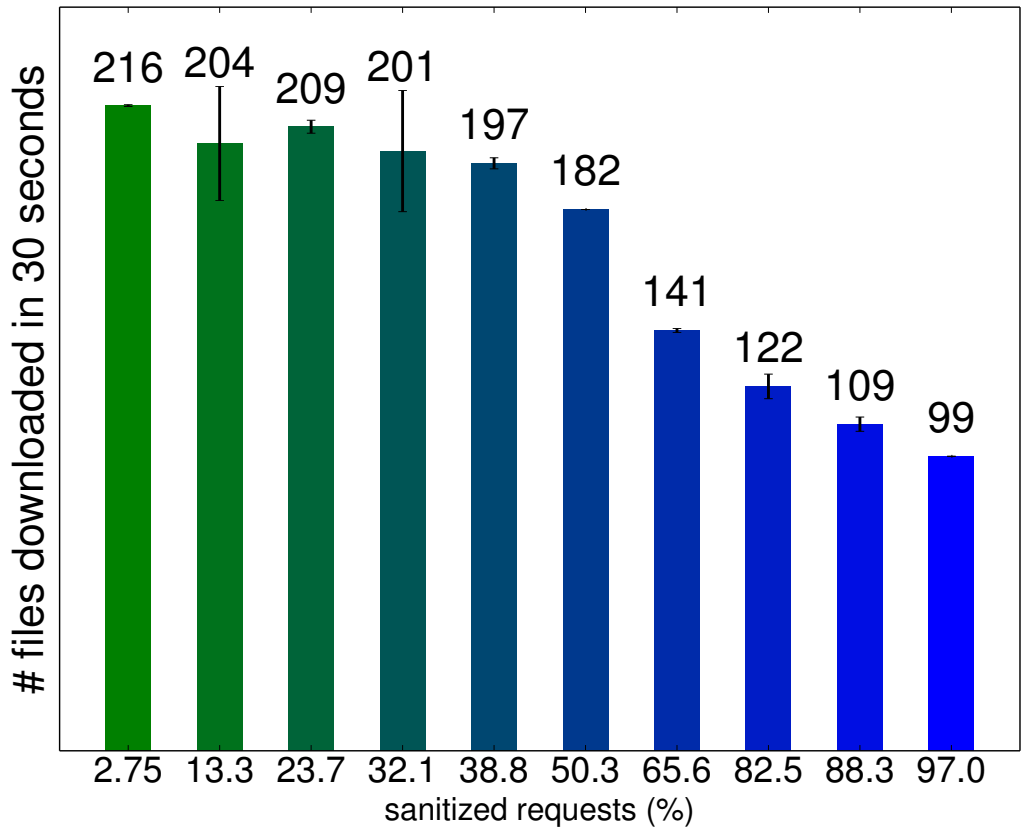
x	y	gcc -O0	gcc -O2
1	2	3	3
1000000000	1000000000	2000000000	2000000000
2000000000	2000000000	2147483647	-294967296

Multi-version execution for security

- Prevents compiler backdoors
- Detects exploits based on undefined behavior
- Interesting programs:
  - Sudo
  - OpenSSH
  - Password vaults
  - GnuPG

Conclusion

- Retain performance? ✓
- Prevent backdoors? ✓
- Still detect bugs?
- Real server software?
- Other analyses?

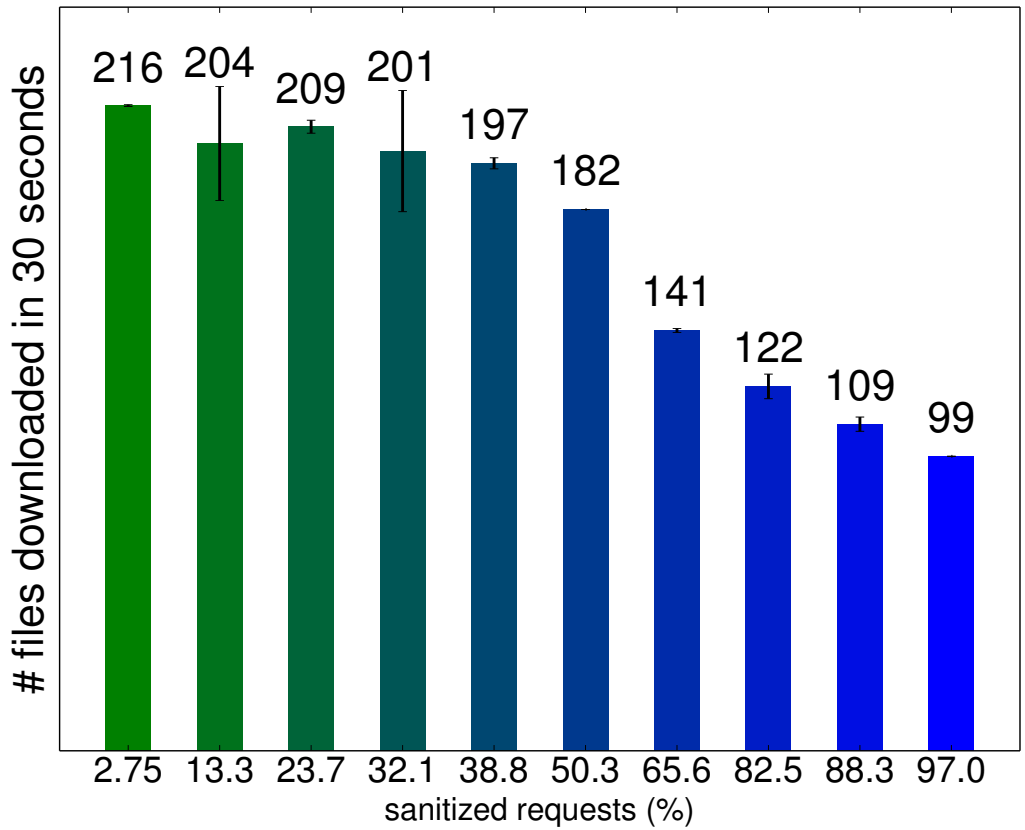


- Luís Pina and Cristian Cadar, *Towards Deployment-Time Dynamic Analysis of Server Applications*, WODA, 2015
- Cristian Cadar and Luís Pina and John Regehr, *Multi-Version Execution Defeats a Compiler-Bug-Based Backdoor*, http://blog.regehr.org/archives/1282, 2015

Thank you!

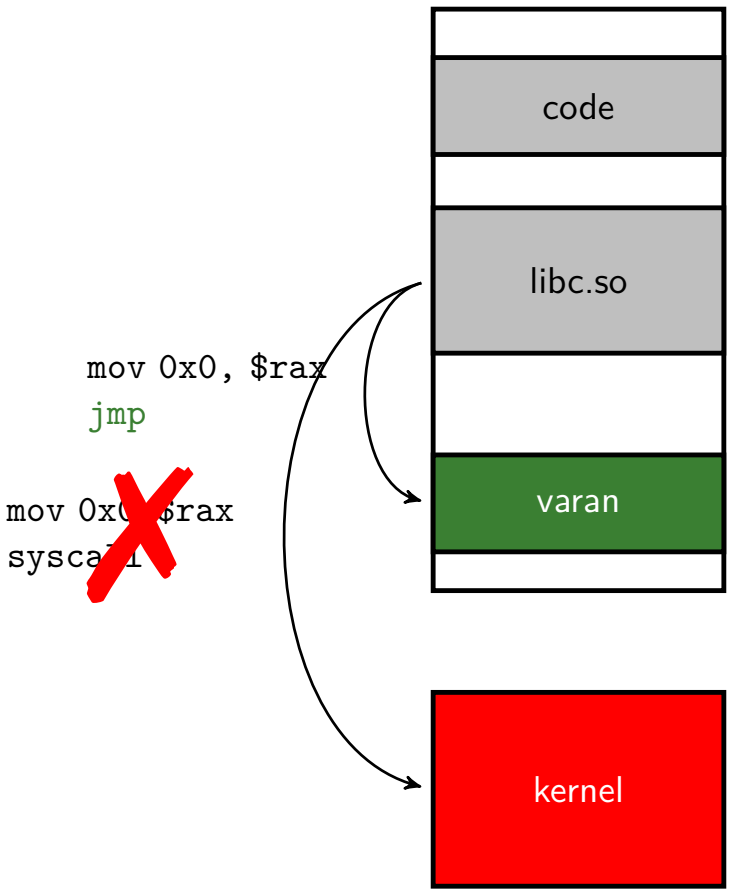
Conclusion

- Retain performance? ✓
- Prevent backdoors? ✓
- Still detect bugs?
- Real server software?
- Other analyses?

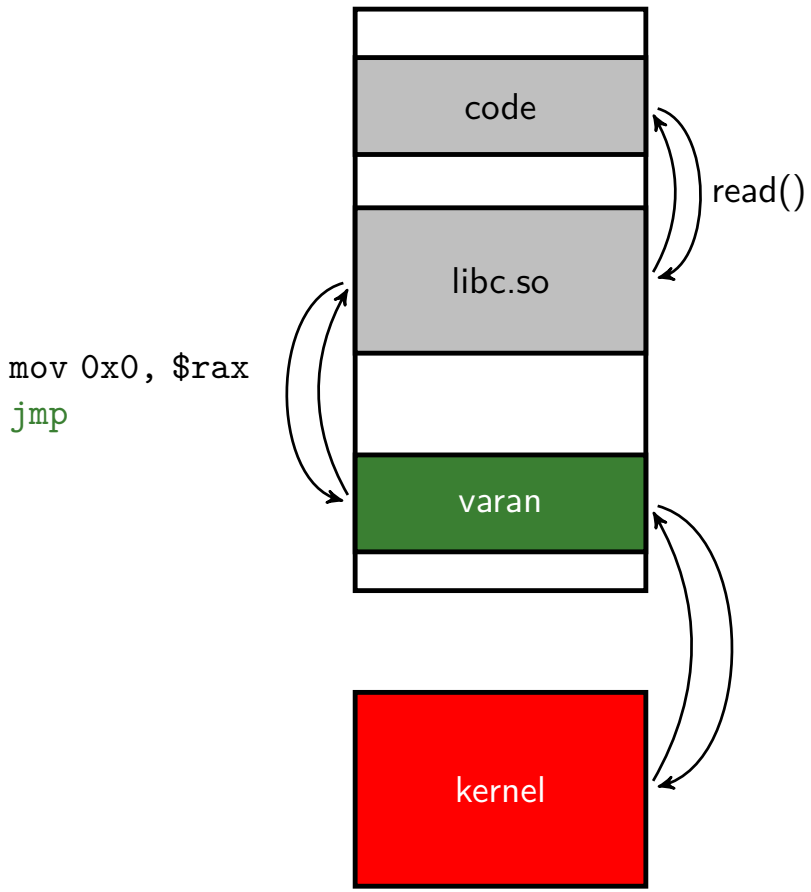


- Luís Pina and Cristian Cadar, *Towards Deployment-Time Dynamic Analysis of Server Applications*, WODA, 2015
- Cristian Cadar and Luís Pina and John Regehr, *Multi-Version Execution Defeats a Compiler-Bug-Based Backdoor*, http://blog.regehr.org/archives/1282, 2015

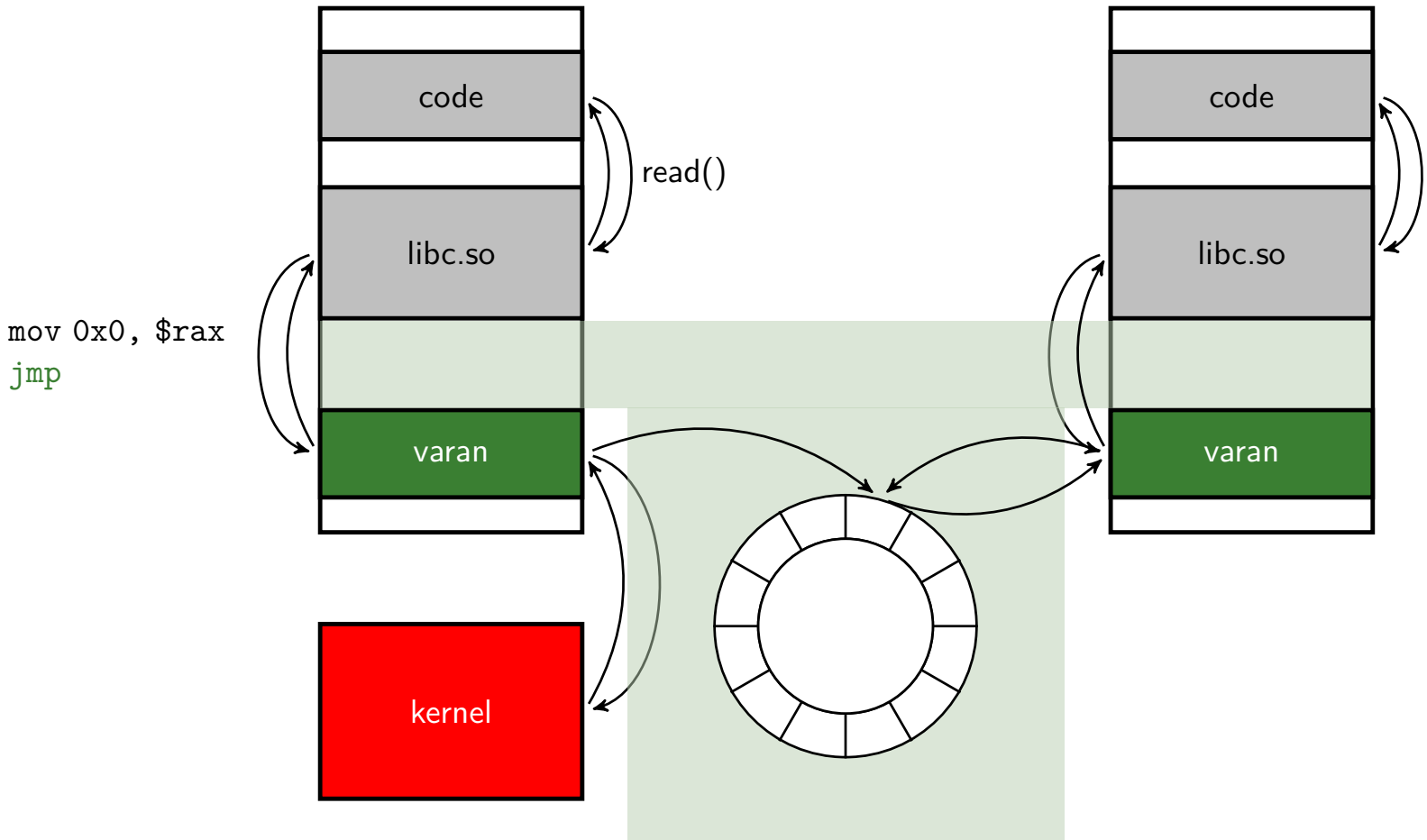
Varan Implementation



Varan Implementation



Varan Implementation



Drop requests — Implementation Challenges

```
01
02      while (true) {
03
04          sckt = accept();          // Wait for client
05
06          req = parse(read(skt)); // Handle request
07          file = open(req);
08          rsp = read(file);
09
10
11
12          write(skt, rsp);          // Send response
13
14      }
15
```

Drop requests — Implementation Challenges

```
01
02      while (true) {
03
04          sckt = accept();            // Wait for client
05
06          req = parse(read(skt));     // Handle request
07          file = open(req);
08          rsp = read(file);
09
10
11
12          write(skt, rsp);            // Send response
13
14      }
15
```

Drop requests — Implementation Challenges

```
01
02      while (true) {
03          open("VARAN_DUMMY");
04          sckt = accept();            // Wait for client
05
06          req = parse(read(skt));     // Handle request
07          file = open(req);
08          rsp = read(file);
09
10
11
12          write(skt, rsp);            // Send response
13
14      }
15
```

Drop requests — Implementation Challenges

```
01 void http_server() {
02     while (true) {
03         open("VARAN_DUMMY");
04         sckt = accept();          // Wait for client
05
06         req = parse(read(skt)); // Handle request
07         file = open(req);
08         rsp = read(file);
09
10         rsp = bzip2(rsp);
11
12         write(skt, rsp);          // Send response
13
14     }
15 }
```